

# Gallery of programmer interfaces

[JonathanMEdwards@gmail.com](mailto:JonathanMEdwards@gmail.com)  
@jonathoda  
<http://alarmingdevelopment.org>

These images bear witness to the passionate work of so many people striving to improve programming.  
Thank you.

Not covered: spreadsheets, data analysis/viz.

Slideshow version at:

<https://docs.google.com/presentation/d/1MD-CgzODFWzdpnYXr8bEgysfDmb8PDV6iCAjH5Jlval/pub?start=false&loop=false&delayms=3000>

## See also

<https://www.cs.cmu.edu/~caitlin/papers/NoviceProgSurvey.pdf>

<http://blog.interfacevision.com/design/design-visual-programming-languages-snapshots/>

<https://github.com/ivanreese/visual-programming-codex>

<https://github.com/shaunlebron/history-of-lisp-parens>

<https://futureofcoding.org/catalog/>



# In the beginning

IBM 402  
Plugboard  
1948



Teletype  
ASR-33  
1963



IBM 026  
Keypunch  
1949



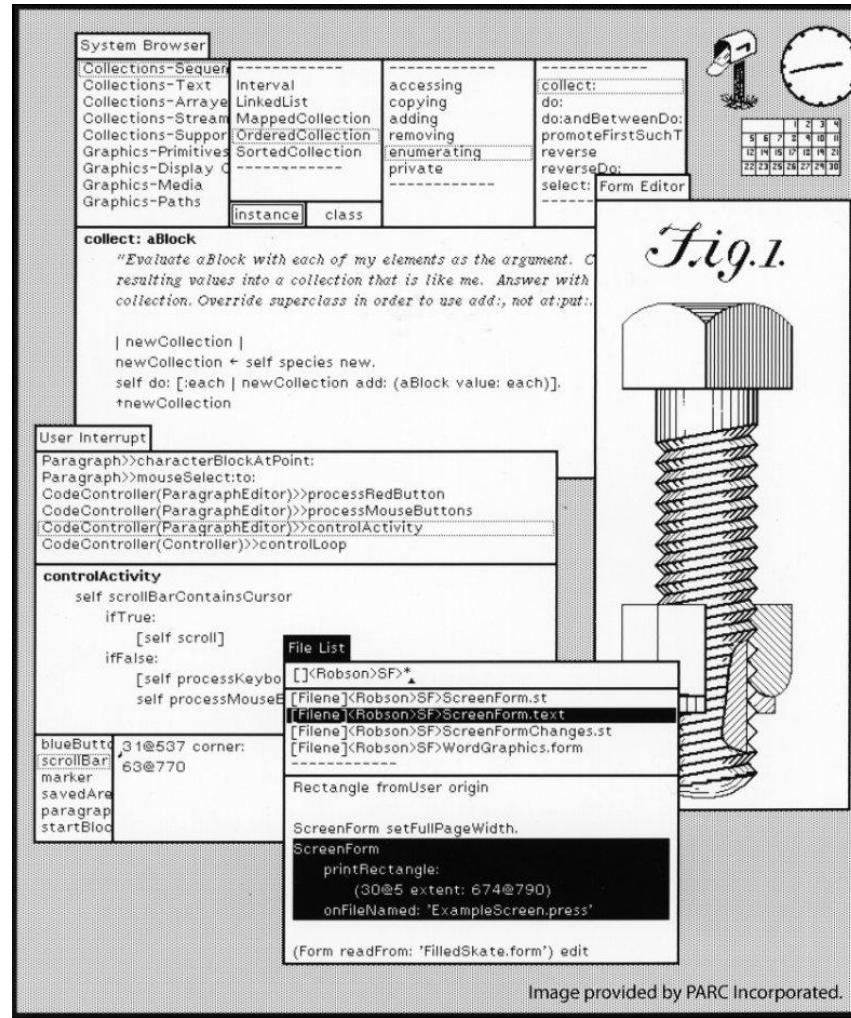
DEC  
VT05  
1970

# Symbolics Lisp Machine 1980

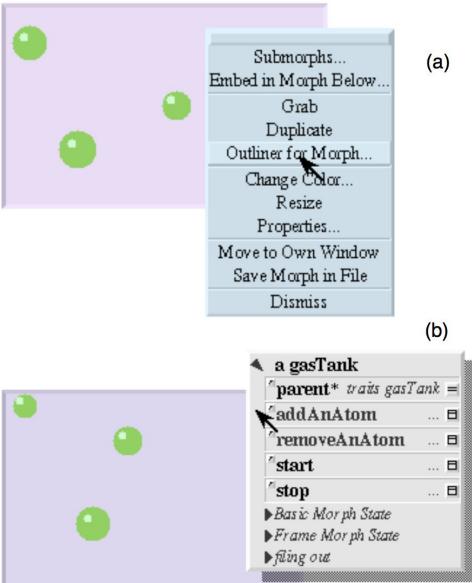
<p><i>Top of object</i></p> <pre>*X2* Value is NIL Function is unbound Property list: (DOCUMENTATION "...") SPECIAL #&lt;UNIX-PATHNAME "VIXEN: //dess//workstyles. Package: #&lt;Package GRAPHICS 36635277&gt;</pre> <p><i>Bottom of object</i></p> <pre>#&lt;Stack-Frame COMPUTE-DENS PC=12&gt; *X2*</pre> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <tr> <td style="width: 50%;">Args: Arg 0 (X): 1800</td> <td style="width: 50%;">Locals:</td> </tr> </table> <p><i>More above</i></p> <pre>(DO-ARROW) (DRAW-ARROW-GRAPHIC 1200 1800 1800) (DRAW-BIG-ARROW) (STRIPE-ARROWHEAD) (COMPUTE-NLINES 1800) (COMPUTE-DENS 1800)</pre> <p><i>More below</i></p> <pre>Return to normal debugger, staying in error context. Supply replacement argument Return a value from the --INTERNAL instruction Retry the --INTERNAL instruction Lisp Top Level in Lisp Listener 1</pre> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <tr> <th>What</th> <th>Error</th> <th>Inspect</th> <th>Return</th> <th>Set arg</th> <th>Retry</th> <th>T</th> </tr> <tr> <td></td> <td>Arglist</td> <td>Edit</td> <td>Throw</td> <td>Search</td> <td></td> <td>NIL</td> </tr> </table> <pre>&gt;&gt;Trap: The first argument given to SYS--INTERNAL, NIL, was not a number.</pre> <p><i>Choose a value by pointing at the value. Right gets object into error handler.</i>  <i>GRAPHICS: Tyl</i></p>	Args: Arg 0 (X): 1800	Locals:	What	Error	Inspect	Return	Set arg	Retry	T		Arglist	Edit	Throw	Search		NIL	<p><i>Top of History</i></p> <pre>*err*</pre> <pre>#&lt;ARROW 10020042&gt; #&lt;Message handler for ARROW&gt; #'(METHOD BASIC-ARROW :COMPUTE-DENS)</pre> <p><i>Bottom of History</i></p> <p><i>Top of object</i></p> <pre>#CARROW 10020042&gt; An instance of ARROW. #&lt;Message handler for ARROW&gt;</pre> <p><i>Top of object</i></p> <pre>DEPTH: 6 TOP-EDGE: 10 TOP-EDGE-2: 5</pre> <p><i>More below</i></p> <p><i>Top of object</i></p> <pre>#&lt;Message handler for ARROW&gt; :COMPUTE-DENS: #'(METHOD BASIC-ARROW :COMPUTE-DENS) :COMPUTE-NLINES: #'(METHOD BASIC-ARROW :COMPUTE-NLINES) :COMPUTE-POINTS: #'(METHOD BASIC-ARROW :COMPUTE-POINTS) :COMPUTE-STRIPE-D: #'(METHOD BASIC-ARROW :COMPUTE-STRIPE-D) :COMPUTE-TOP-EDGES: #'(METHOD BASIC-ARROW :COMPUTE-TOP-EDGES)</pre> <p><i>Top of object</i></p> <pre>#&lt;DTP-COMPILED-FUNCTION (METHOD BASIC-ARROW :COMPUTE-DENS) 48660073&gt; 0 ENTRY: 4 REQUIRED, 0 OPTIONAL 1 PUSH-INDIRECT *D1* 2 PUSH-INDIRECT *D2* 3 PUSH-INDIRECT *D1* 4 BUILTIN --INTERNAL STACK 5 PUSH-LOCAL FP13 ;X 6 PUSH-INSTANCE-VARIABLE 2 ;PBK 7 BUILTIN --INTERNAL STACK 18 PUSH-INSTANCE-VARIABLE 15 ;X2 11 PUSH-INSTANCE-VARIABLE 2 ;PBK 12 BUILTIN --INTERNAL STACK 13 BUILTIN FLOAT STACK 14 BUILTIN /=INTERNAL STACK 15 BUILTIN *=INTERNAL STACK 16 BUILTIN *=INTERNAL STACK 17 RETURN-STACK</pre> <p><i>Bottom of object</i></p> <pre>Choose a value by pointing at the value. Right finds function definition. GRAPHICS: Tyl</pre>
Args: Arg 0 (X): 1800	Locals:																
What	Error	Inspect	Return	Set arg	Retry	T											
	Arglist	Edit	Throw	Search		NIL											

Figure 8. The Window Debugger: inspecting the variable \*x2\*.

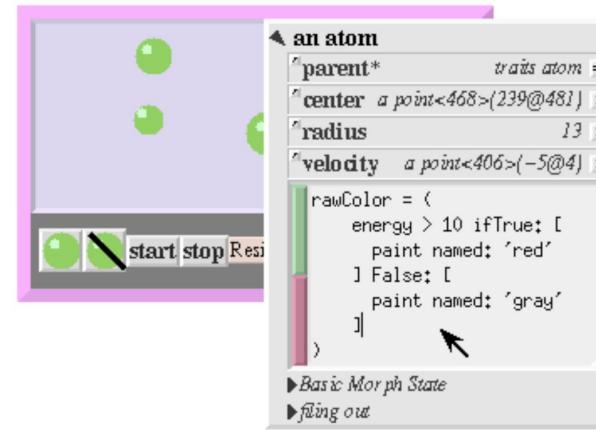
# Smalltalk-80



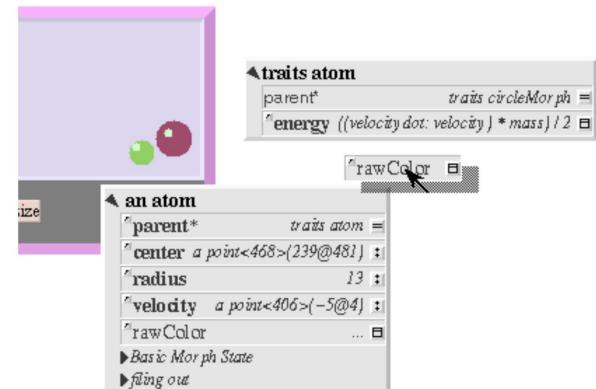
# Self 1987



**Figure 4.** In a Self window, the user pops up the meta menu on the ideal gas simulation (a). Selecting “outliner” gives the Self-level representation to the user, which can be carried and placed as needed (b). (The italic items at the bottom of the outliner represent slot categories that may be expanded to view the slots. Unlike slots, categories have no language level semantics and are essentially a user interface convenience.)



**Figure 9.** The user has selected one atom on which to experiment. The user changes the “rawColor” slot from a computed to a stored value by editing directly in the atom’s outliner.



**Figure 10.** The user copies the modified rawColor slot as a first step in getting the computed method of coloring more widely shared. Because slots can be moved about readily, restructuring changes are relatively light weight, enhancing the sense of flexibility.

# Boxer 1986

You can make arcs with the procedures ARCREIGHT and ARCLEFT.

```

arcleft [input size degrees
repeat degrees
  forward size
  left 1]
arcright [input size degrees
repeat degrees
  forward size
  right 1]

```

Here are some procedures that draw pictures using arcs:

```

flower [sun r
  slinky r
  sun input r
    ray repeat 2
      arcleft r 90
      arcright r 90
    repeat 9
    ray
    right 160]

```

**Graphics**

Try running some of the following commands to draw pictures in the graphics box above.

```

sun R: 0.7
flower S: 1.4
slinky R: 2.6

```

Graphics appear inside regions of the screen called graphics boxes. The box labeled SUN defines the procedure that drew the design in the graphics box above. Definitions (boxes with name tabs on them) can be invoked anywhere within the box in which they appear. This scoping rule provides a general block-structuring capability that has been used here to include RAY as an internal subprocedure of SUN. The bottom line of the screen is a menu from which the user can select

commands to be run. Here, as indicated by the small arrow (mouse cursor), the user has selected SUN with an input of 0.7. The letter R followed by a colon is a prompt generated by the system to indicate that the SUN procedure requires one argument, named R. Such prompts are generated automatically whenever users type the name of a defined procedure and press a help key. In general, any text on the screen can be modified or run.

**phone-book**

**list**

Data	Data	Data	Data
name	Aunt Tillie	address	43 2nd St Bellwood
phone	SSS-1212		

**function-1-key**

```

for x in list
  if x.name = name
    change number x.number

```

If you put a name in the NAME box, and press the FUNCTION-1 key the phone number will appear in the NUMBER box.

**number** Data      **name** Data Uncle Hiram

This box is a utility for finding phone numbers. Its local database is a box called LIST that contains subboxes with a

standard format. The procedure named FUNCTION-1-KEY will be run whenever the FUNCTION-1 key is pressed.

**Phone-book**

**list**

Data	Data	Data	Data
name	Data	address	Data
phone	Bellwood		

**function-2-key**

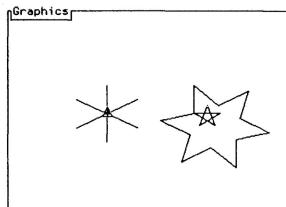
Data	Data	Data	
name	Data	phone	Data

If you put a name in the NAME box, and press the FUNCTION-2 key the phone number will appear in the PHONE box.

**number** Data      **name** Data

If you put a name in the NAME box, and press the FUNCTION-2 key the phone number will appear in the PHONE box.

the template and label it with the name FUNCTION-2-KEY. (The template shown here is especially useful if you happen to know many people who live in Bellwood whose phone numbers begin with 555.)



**star** [input size angle
repeat 360 / angle
 step size
 right angle]

```

tell minnie star SIZE:40 ANGLE:60
tell mickey star SIZE:40 ANGLE:60

```

This graphics box contains two sprites—Minnie (the small triangle) and Mickey (the small five-pointed star). Each sprite has been told to run the procedure called STAR. The designs drawn are different, because each sprite has its own definition of the STEP procedure when executing STAR.

FIGURE 5a. A Graphics Box

**Graphics**

**Mickey**

Data	Data	Data	
XPOS	0.00006	YPOS	0.0
HEADING	0	step	[input size   forward size   right 135   forward size   left 135]

**shape**

```

right 90
repeat 5
  forward 30
  right 144

```

**minnie**

Data	Data	Data	
XPOS	-50.0	YPOS	0.0
HEADING	0	step	[input size   forward size   back size]

**m-click**

```

forward 30

```

Here is the graphics box of Figure 5a, shown in data form so that all of its computational structure is visible, changeable, and extendable in ordinary Boxer textual format. The XPOS, YPOS, and HEADING variables show the position and head-

ing of each sprite. In addition, Mickey has a SHAPE procedure that makes him appear as a five-pointed star; Minnie has no SHAPE procedure and hence appears in the default shape, a triangle.

# HyperCard 1987

**Welcome to HyperCard**

©1987-1990 Apple Computer, Inc.  
All Rights Reserved.

Links: HyperCard Tour, HyperCard Help, Practice, New Features, Art Bits, Addresses, Phone Dialer, Graph Maker, QuickTime Tools, AppleScript Mail Merge, AppleScript Text Controls.

**SmallCalculator**

**Design Notebook I**

**Button Positive Gravity**

Group Positive Acceleration  
Composed in project launch 8/14/92 9:38 AM

**Show Me**

**Appearance**  **Behavior**  **How it works**

Triggers  MouseUp  MouseStillDown  MouseEnter  MouseLeave  
Actions for this Trigger Accelerated Motion 1  
Stop increasing 1  
Display a Value 1  
Display a Value 1

Variables for this Trigger time.Display a Value:{number to display}:1  
location.Display a Value:{number to display}:1  
velocity.Display a Value:{number to display}:1

Behavior

```

put the top of card button "Positive Gravity" into card field "Position" — Save Top of Object 1,1
put the left of card button "Positive Gravity" into card field "Move over" — Save the left of button 1,1
choose eraser tool — Clear the Graphics,1
doMenu "Select All" — Clear the Graphics,2
choose browser tool — Clear the Graphics,3
doMenu "Clear Picture" — Clear the Graphics,4
put card field "Position" into location — Accelerated Motion 1,1
put card field "Starting Velocity" into velocity — Accelerated Motion 1,2
repeat (time - 1 to 999) — Accelerated Motion 1,3
    add 2 to velocity — Accelerated Motion 1,4
    add velocity to location — Accelerated Motion 1,5
    put time into card field "Time" — Display a Value 1,1
    Created 7/31/92 by Mark Gudzial in project Gravity Simulation
  
```

**SmallCalculator**

**Button Info**

Button Name: =

Card button number: 15   Style: Round Rect

Card part number: 16   Family: None

Card button ID: 19

Preview

Checkboxes:  Show Name,  Auto Hilite,  Enabled

Buttons: OK, Cancel, Script..., Contents..., Tasks..., LinkTo..., Icon..., Text Style...

**Script of card button id 19 = "**

Scripting language: HyperTalk

Handlers:

Length: 103

Functions:

```

on mouseUp
    get the value of card field "lcd"
    put the value of it into card field "lcd"
end mouseUp
  
```

**Fig. 23.** An editor for the Positive Gravity Button. When the mouse goes up, Emile will execute 4 actions: Accelerated Motion 1, Stop Increasing 1, and Display a Value 1 (2 times). At the bottom of the screen, we can see the code that Emile will execute. Underlined text corresponds to parameters (or slots) that the user can fill in using menu options and dialog boxes.

# Alice Pascal 1987

We were going to tell  
you all about

```
program Alice;
begin
  writeln('Hello World');
end.
```

You program from "templates"

```
program Alice;
begin
  writeln('Hello World');
end.
```

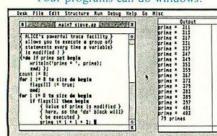
Multiple window editing



Over 700 help screens



Your programs can do windows!



"If I needed to learn Pascal all over again, or were going to teach a course in the language, I can't imagine using any program other than ALICE."

— Adam Green, Infoword

Visit your dealer or send us \$79.95 (USD)  
plus \$5 for shipping and handling. VISA  
and MasterCard orders may phone collect.  
Dealer inquiries invited.

Trademarks: Turbo Pascal — Borland International, IBM-PC — IBM, Atari ST — Atari Corp., GEM — Digital Research Inc. ALICE refers to ALICE: The Personal Pascal, a trademark of Graham Software Corp., used with permission.



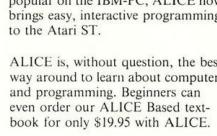
but we'd rather show  
you . . .

```
program Alice;
begin
  writeln('Hello World');
end.
```

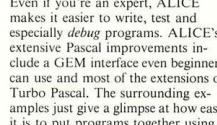
Instant-run Pascal interpreter

```
program Alice;
begin
  writeln('Hello World');
end.
```

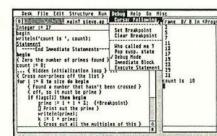
Every error fully explained



Easy graphics in windows



Simple menus and events



Breakpoints, Single Step

"ALICE may be the most advanced programming environment currently available for the PC."

— Michael Covington, PC World

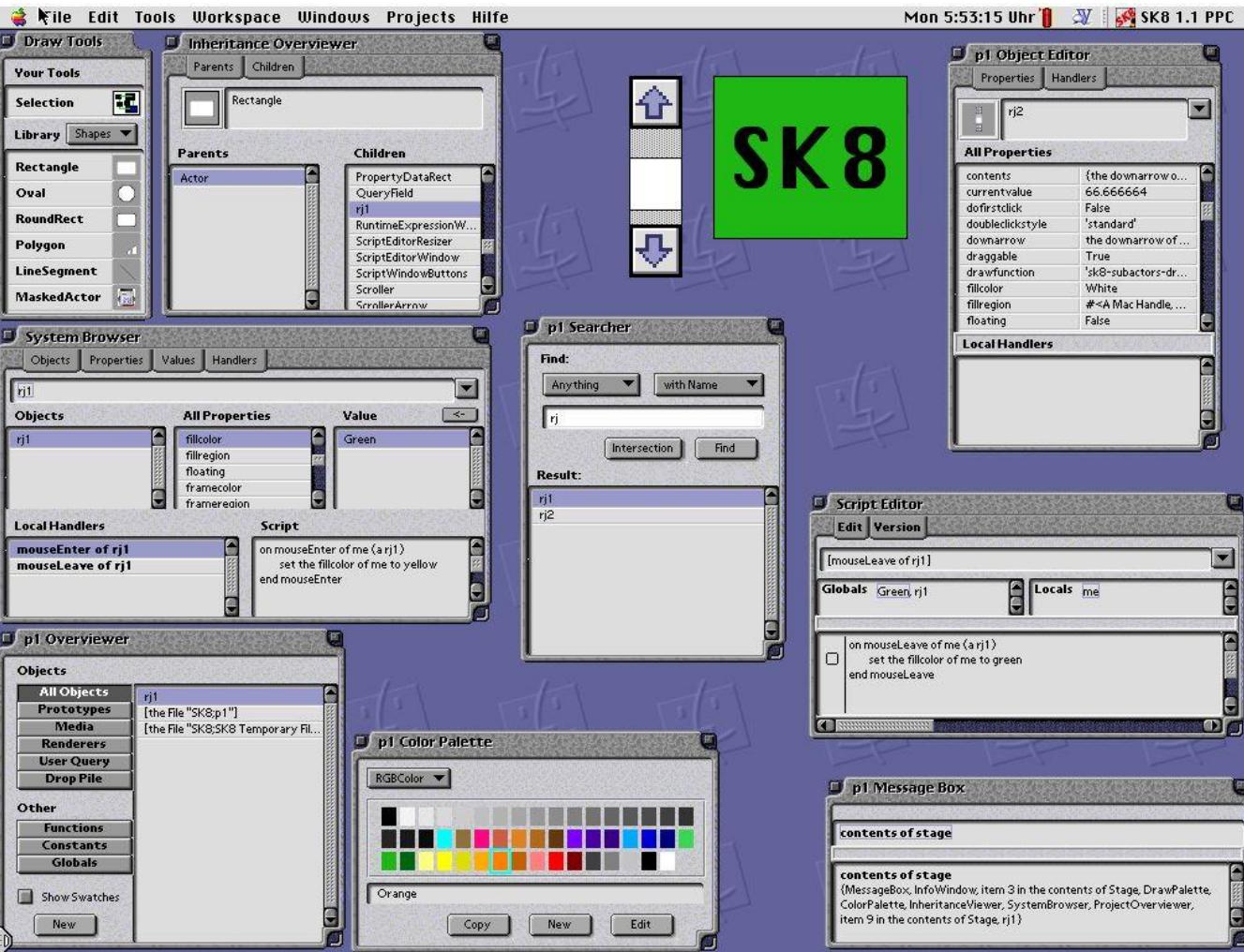
Looking Glass Software Limited  
124 King St. N. Waterloo, Ontario  
N2L 2X8  
519/884-7473

"If you enjoy programming languages, this comes pretty darn close to being as much fun as a video game."

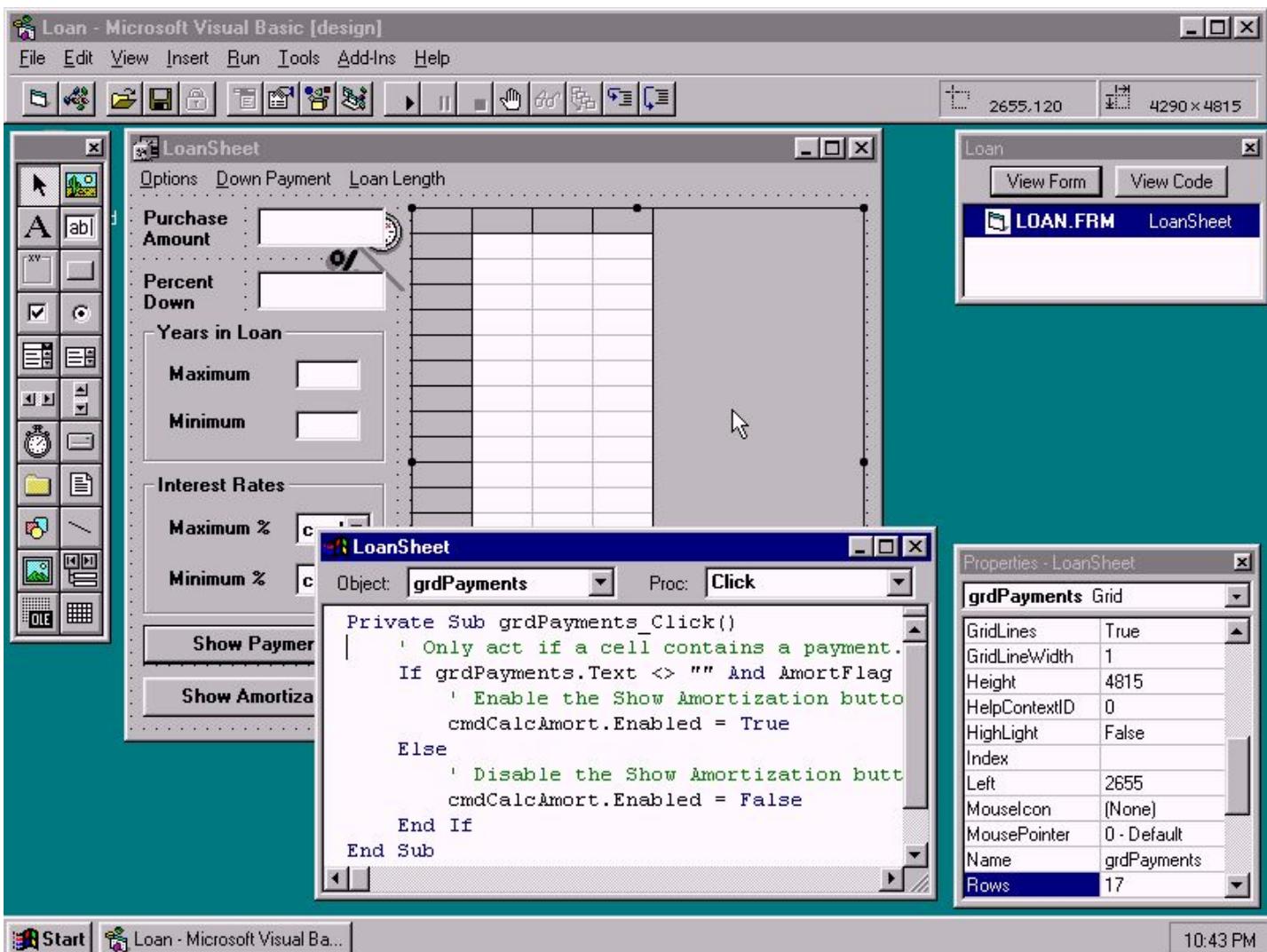
— Gene Wilburn, Computing Canada

Looking Glass Software  
looking glass software

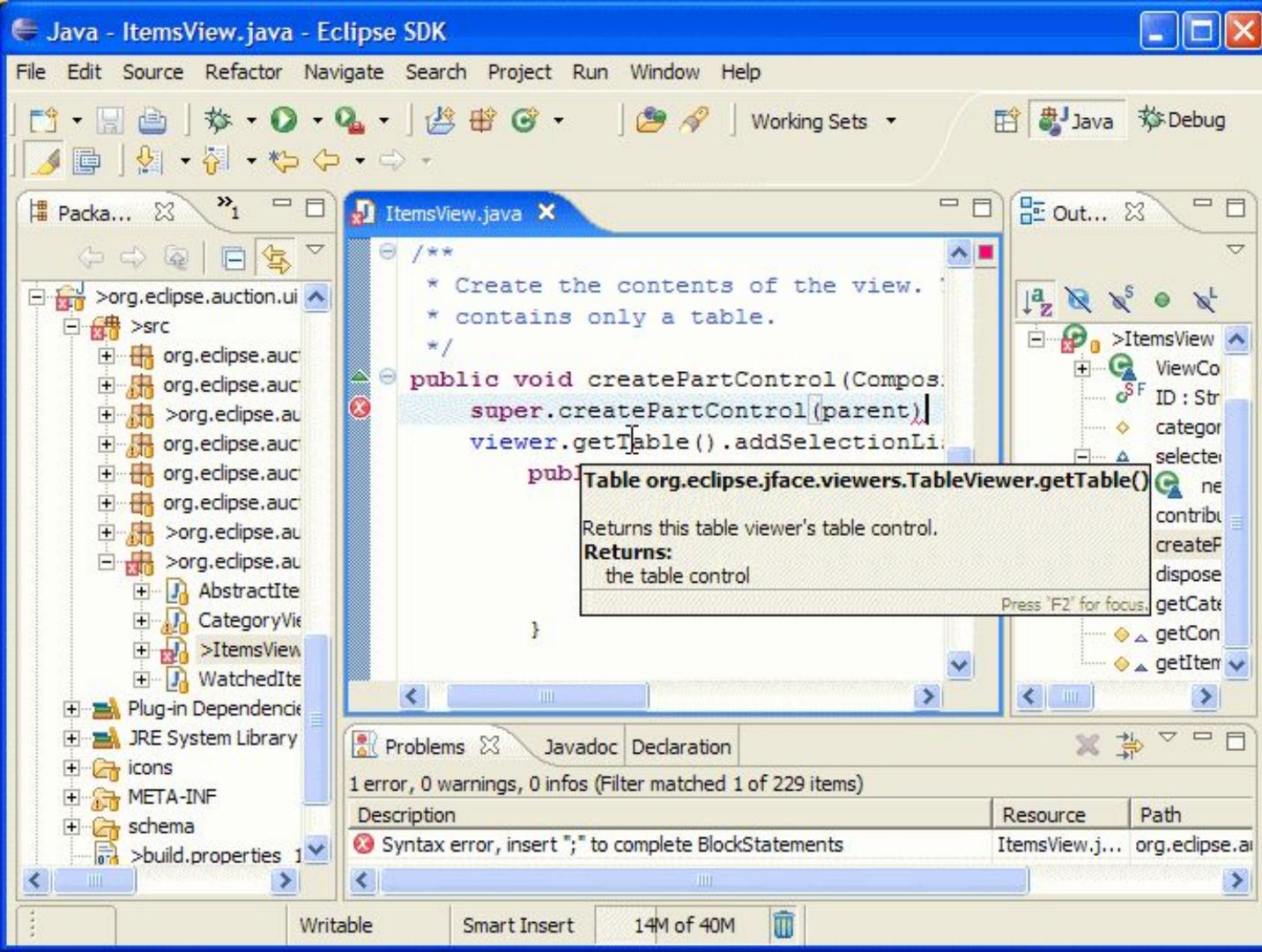
SK8 1994



# Visual Basic 1991



# Eclipse 2001



# Squeak/eToys

## 1996

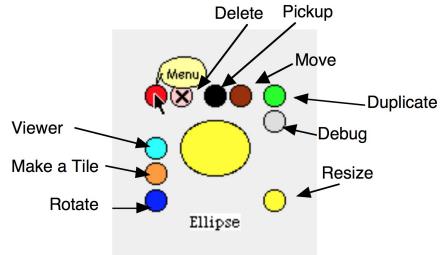


Figure 8: Standard Morphic Halos

The screenshot displays several windows from the Squeak/eToys environment:

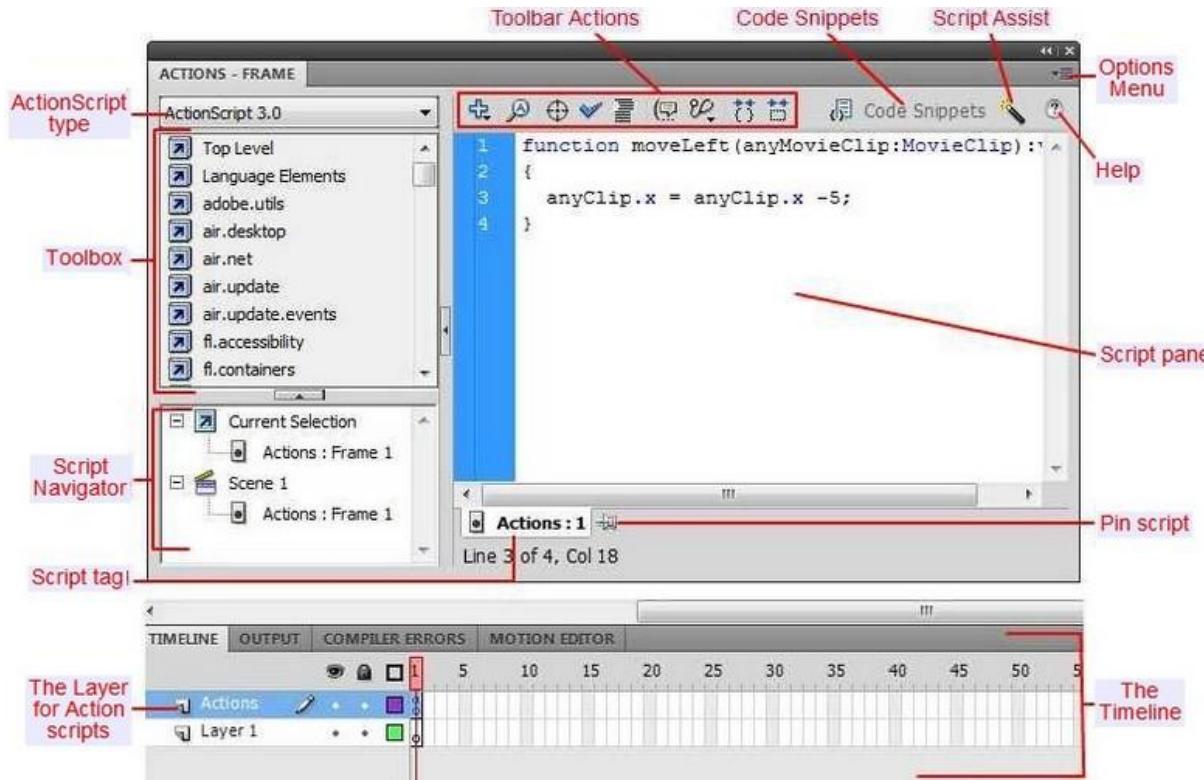
- scripting area:** Shows a green background with a blue wavy line and a pink circle. A message "Squeak Smalltalk 3.0 is coming!" is displayed.
- Explorer:** A hierarchical tree view of objects. One node is highlighted in red: "2: a SystemWindow('Mister Postman' 2897)".
- System Browser:** Shows a list of categories and sub-categories under "Morphic-Examples".
- Mister Postman:** A window showing an email message from "giovanni.g@infinito.it" to "Back to the Future" with the subject "Back to the Future".

At the bottom, there are several buttons: "stop", "step", "go", "clear", "clsar", and "instance". A note at the bottom explains the behavior of the "Morphic-Bounce" example.

The screenshot shows the "Your Own Lunar Lander Game" project:

- Code Blocks:** Several green script blocks are shown, each with conditions and actions. One block tests "Joystick2's upDown > 0" to move a "flame" object.
- Preview Window:** A 3D rendering of a lunar landscape with a blue planet in the background. A red rocket ship is on the surface, and a small flame is visible.
- Buttons:** On the right, there are buttons for "stop", "step", "go", "ship reset", and "ship's ySpeed = 0.0".

# Flash MX 2003



# Scratch 2005

The image shows the Scratch 2005 interface with a project titled "Slime Ninja" by DarkLava. The stage features a blue background with clouds, a central "PLAY" button, and a "EXTREME" button. A black ninja sprite is positioned in the center, holding a sword. To the left is a blue slime sprite, and to the right is a green slime sprite. The script editor on the right contains two scripts for the Global sprite:

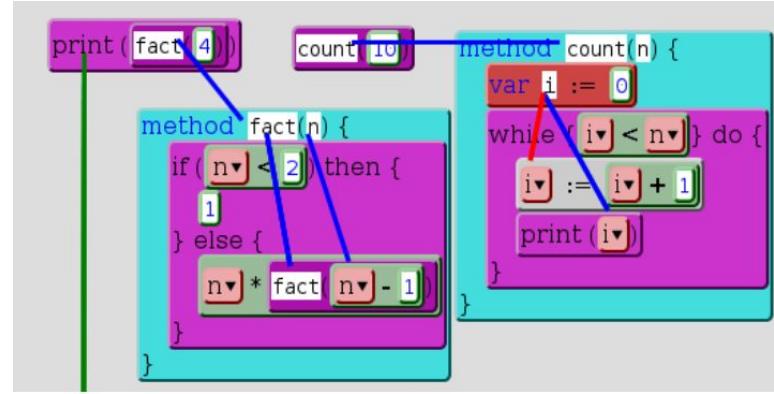
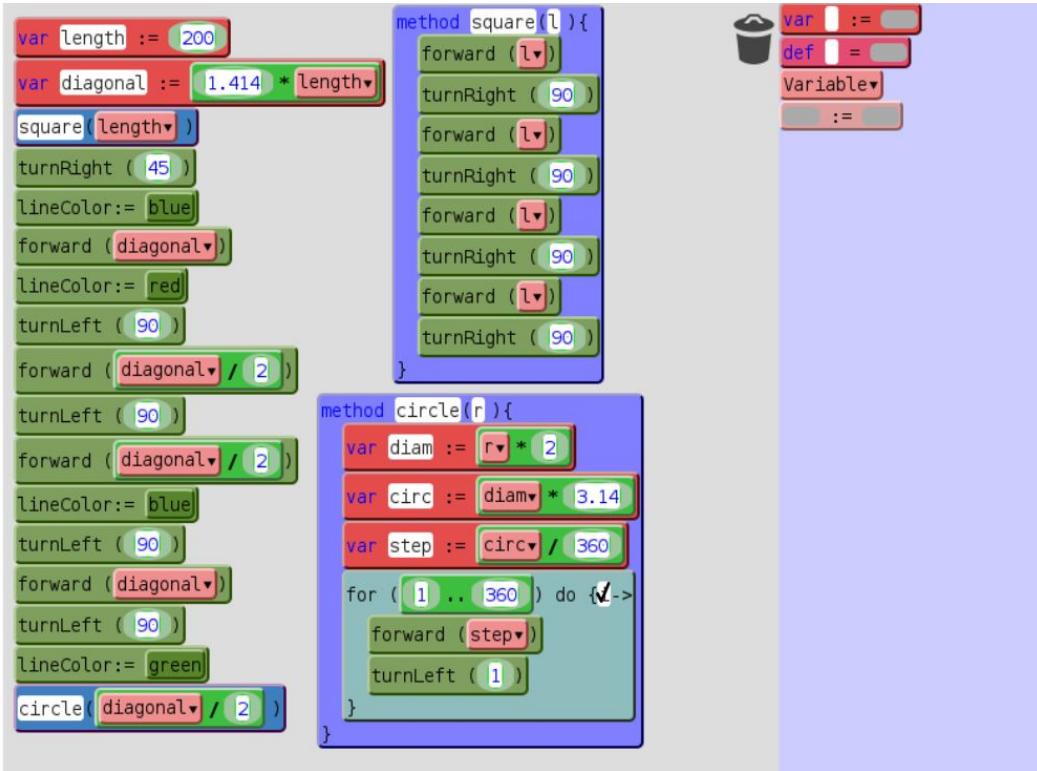
```
when I receive Start
hide
reset timer
repeat until [timer > 15]
  set [Level v] to [0]
  set [Type v] to [0]
  if [pick random 1 to 2 = 1] then
    set x to [240]
  else
    set x to [-240]
  set x to [((x - x position) / 800)]
  set y to [-135]
  set [Y-vel v] to [0]
  create clone of [myself v]
  wait [pick random 1.5 to 2.5] secs
end
repeat until [timer > 45]
  set [Type v] to [0]
  if [pick random 1 to 2 = 1] then
    set x to [240]
  else
    set x to [-240]
  set x to [((x - x position) / 800)]
  set y to [-135]
  set [Y-vel v] to [0]
  create clone of [myself v]
  wait [pick random 1.5 to 2.5] secs
end
```

On the left, there are two additional scripts:

- A script for the Stage that repeats until the distance to mouse-pointer is greater than 50 steps, moving 10 steps.
- A script for a Global sprite that plays a note at Middle C (60) for 0.5 beats when the sprite is clicked.

The bottom left shows the sprite and backdrop library, and the bottom right shows the stage and script controls.

# Tiled Grace 2014



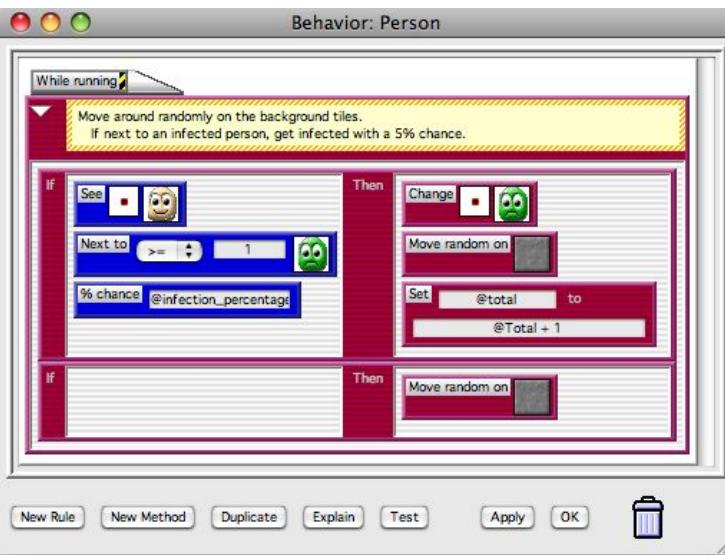
Only Number can go here, not String.

```

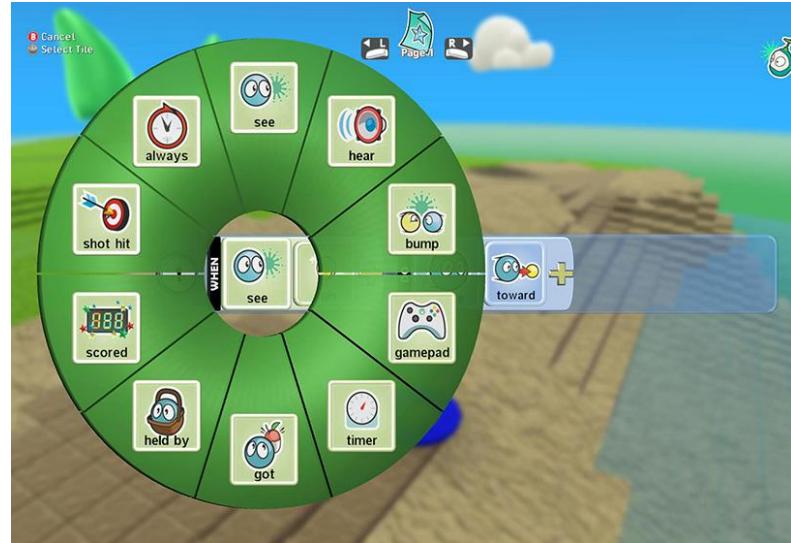
[+ "Hello, world!"]

```

# AgentSheets 1991, AgentCubes 2006



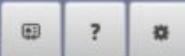
# Kodu 2009



# Gidget 2014

## Level 4. Helping Gidget Take Objects to Different Locations

total bonus \$0.30 |



### code & goals

```
left  
up  
grab  
drop /goop/  
down 3  
left 1  
grab /puppy/  
left 2  
up 3  
grab /goop/  
right 4
```

restore original code

1 program written in imperative language

### world



4 visualization of world state

3 executable goals

2 execution controls

### gidget

```
energy  
grabbed  
image  
labeled  
layer  
name  
position  
rotation  
scale  
transparency
```

5 visualization of program state or assessment question area

6 explanation of program execution

grab: Which **object** do you want me to grab?

# HANDS 2002

rose		tulip		orchid		bumble	
name	value	name	value	name	value	name	value
cardname	rose	cardname	tulip	cardname	orchid	cardname	bumble
x	208	x	350	x	490	x	636
y	80	y	80	y	80	y	80
group	flower	group	flower	group	flower	group	bee
nectar	100	nectar	150	nectar	75	nectar	0

Figure 3. When the system evaluates the query *all flowers* it returns *orchid, rose, tulip*.

- **all flowers** evaluates to **orchid, rose, tulip**
- **all bees** evaluates to **bumble**
- **all snakes** evaluates to **the empty list**
- **all (flower and (nectar < 100))** evaluates to **orchid**
- **set the nectar of all flowers to 0**
- **FirstItem of all flowers** evaluates to **orchid**
- **GreatestItem in nectar of all flowers** evaluates to **150**
- **Sorted nectar of all flowers** evaluates to **75,100,150**
- **CardWithLeast nectar of all flowers** evaluates to **orchid**
- **Sum nectar of all flowers** evaluates to **325**

**with all flowers**

    set the item's nectar to random from 50 to 100  
end with

**with all flowers calling each the flower**

    set the flower's nectar to random from 50 to 100  
end with

- if f=tulip then // *if-then-else style*  
        set f's nectar to 0  
        otherwise set f's nectar to 100  
end if
- if f=... // *case statement style*  
        tulip then set f's nectar to 0  
        orchid then set f's nectar to 10  
        otherwise set f's nectar to 20  
end if
- if // *cond style*  
        f's nectar > 30 then set f's nectar to 0  
        f's nectar < 30 then set f's nectar to 10  
        otherwise set f's nectar to 20  
end if

# Barista 2006

```
public static boolean isFruit(Shape s)
{
    return s.isRound() || s.isOblong();
}

public static boolean isFruit(Shape s)
{
    return s.isRound() && s.isRed() || s.isOblong() && s.isYellow();
}
```

Figure 10. On top, a “match form” representation of a Boolean expression [15]; on bottom is the more editable view that is shown when the caret is in the expression.

```
Image apple = load("apple.png");
paint(apple, 40);

else if ((right - left) < width && (bottom - top) < height)
{
    Image banana = load("banana.png");
    paint(banana, 40);
}
else
{
    Image bowlOfFruit = load("bowlOfFruit.png");
    paint(bowlOfFruit, 40);
}

Paints a fruit based on the shape supplied.

public void paintFruit(Shape shape)
{
    int left = shape.minX();
    int top = shape.minY();
    int right = shape.maxX();
    int bottom = shape.maxY();

    if ((shape.round() && shape.red()) || (shape.round() && shape.green()))
    {
        Image apple = load("apple.png");
        paint(apple, 40);
    }
}
```

Figure 11. A focus + context interaction that allows users to partially collapse blocks of Java code.

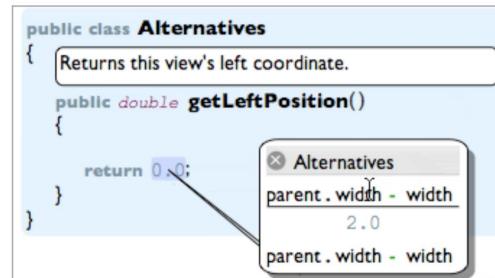


Figure 12. A tool for managing alternative expressions, helping users track and explore different solutions.

# Whyline

2009

Resume Stop ? Why... Undo Redo Your world is paused.

The screenshot shows a game world with a yellow ghost and several red dots. A script editor on the right contains a script for the ghost:

```

When Pac is within 1 meter of Big Dot becomes true
Do this once, when it becomes true
  Do in order
    Big Dot set isShowing to false more...
    Big Dot.isEaten set value to true more...
  
```

**Pac's details**

**properties**: current direction = forward  
create new variable  
capture pose  
color = yellow  
opacity = 1 (100%)  
vehicle = World

**methods**: World.move Pac

**questions**: No variables

**World.move Pac** No parameters

No variables

**script editor**

```

Pac move Pac.current direction 3 meters duration = 1 second style = gentle more...
If both Pac is within 2 meters of Ghost and not Big Dot.isEaten
  Pac resize 0.5 more...

```

**Question: Why didn't Pac resize 0.5?**

**Answer:**

One or more of these actions prevented Pac resize 0.5 from happening. Try following the arrows and checking each action to find out what went wrong.

**runtime actions**

3.821010  
Big Dot.isEaten set to true → true → isEaten → true → Pac is within 2 of Ghost → true → and → not → false → Doing else → false

**causality arrows**

**Questions I've asked**

3.854011

**time cursor traverses execution history**

Annotations highlight various features:

- further questions can be asked
- camera focuses on subject of question
- tooltips show properties' current values
- access to previous questions and answers
- code related to the selection is highlighted

draggable

Copies: [ ]

Add Field

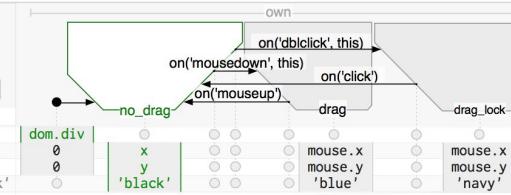
prototypes (div)

x 313

y 763

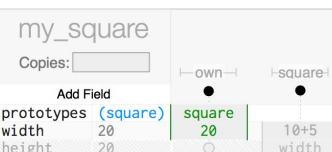
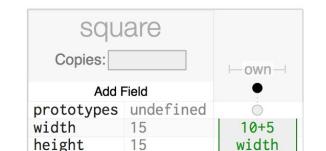
fill black

...



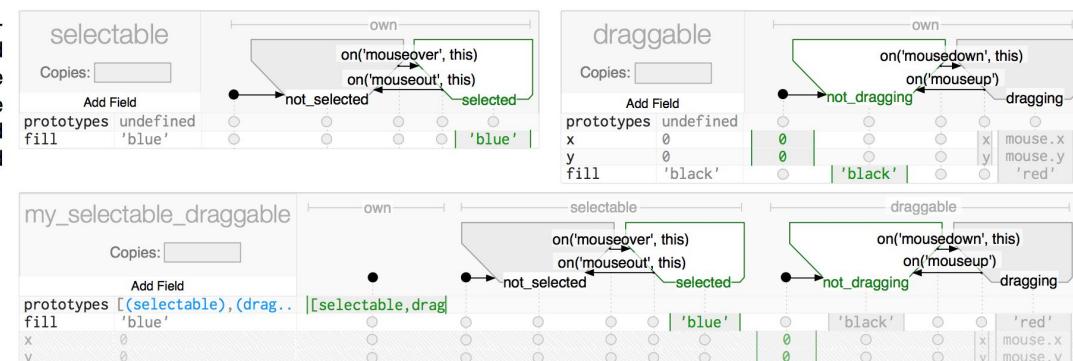
# InterState 2014

**Figure 1:** A basic InterState object, named draggable, which implements draggable and drag lock behaviors. Properties that control draggable's display are represented as rows (e.g. x, y, and fill). States and transitions are represented as columns (e.g. no\_drag and drag). An entry in a property's row for a particular state specifies a constraint that controls that property's value in that state. Here, while draggable is in the drag state, x and y will be constrained to mouse.x and mouse.y respectively, meaning draggable will follow the mouse.



**Figure 3:** InterState uses a prototype-instance inheritance model with multiple inheritance. Prototypes are simply specified in the prototypes property. Here, my\_square inherits from square. Because my\_square does not define a value for height, it inherits the definition of square.height, as indicated by the greyed out text in the columns on the right. Note that my\_square inherits the definition of height, not the value. Thus, its width property evaluates to a different value (20) than it does in square (15).

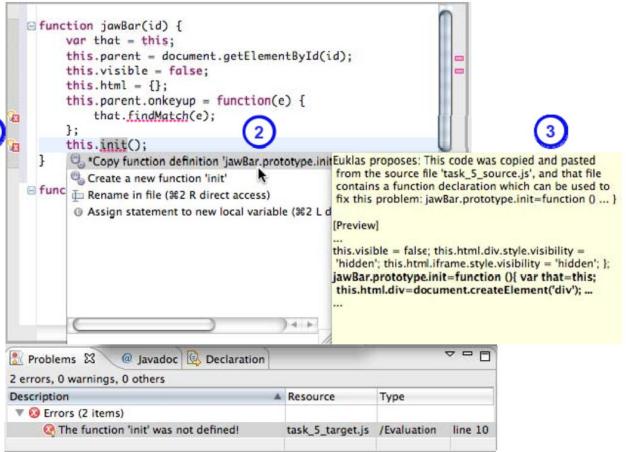
**Figure 4:** An object that inherits from both draggable and selectable behaviors. Note that the definitions for the color property are inherited from draggable ('red') and selectable ('blue').



**Figure 5:** An object with multiple copies; copies is set to ['Jane', 'Sue']. Every copy has two properties: my\_copy, which is set to that copy's item (here, either 'Jane' or 'Sue') and copy\_num, which is set to that copy's index. Here, we are looking at the first copy.

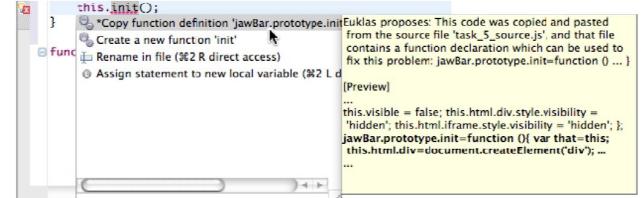
proto  
x undefined  
other\_shape undefined  
other\_shape.x+5

**Figure 6:** Syntax and runtime errors are highlighted in the editor but do not prevent the program from running.

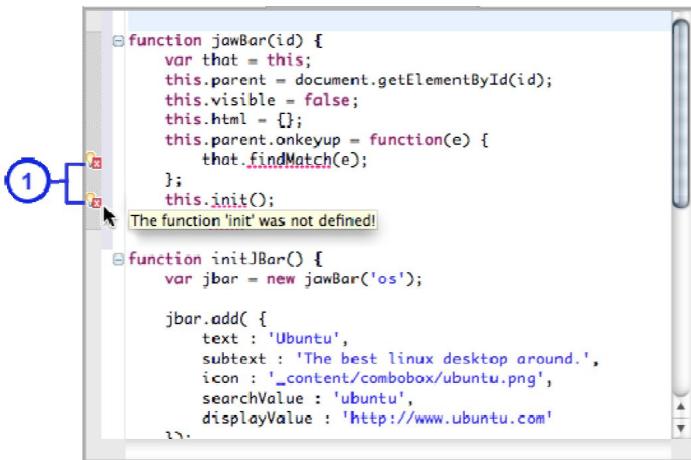


**Figure 1.** Euklas supports peoples' integration attempts as it enhances Eclipse's JavaScript editor by (#1) highlighting errors in the source code, (#2) providing quick fixes including the use of the context in the origin of any copied code, and (#3) including explanations for copy-and-paste errors based on the code of the example snippet that was used.

## Euklas 2014



**Figure 3.** Euklas proposes the top two quick fixes for the undefined function init(). More information about the first (selected) proposed fix is shown in the beige pop-up at the right. The pop-up shows a short proposal explaining what Euklas intends to do, followed by a preview of how the code will look after the selected fix has been applied. Euklas augments Eclipse's quick fix feature which is well-known by many programmers.



**Figure 2.** Euklas marks errors in the function jawBar() after Jamie pastes it into the target file.

# Linked Editing

2004

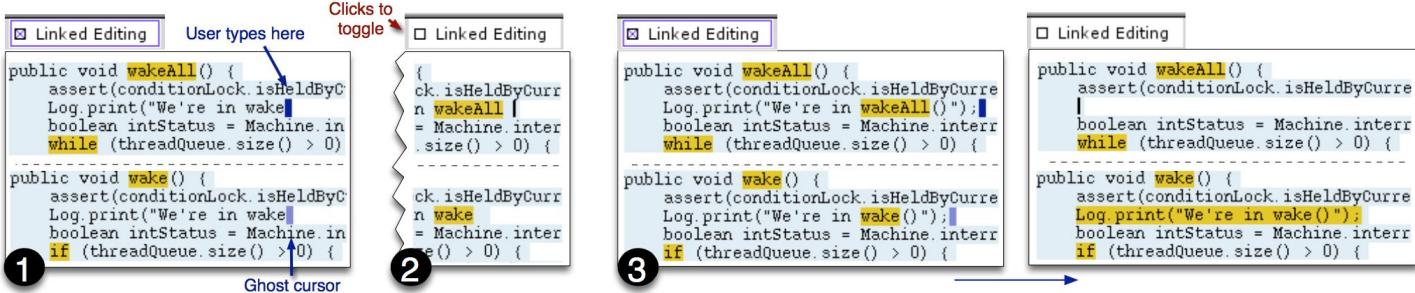


Figure 2. (1) Adding a line to two clones. (2) Modifying one instance. (3) Deleting line in one instance.

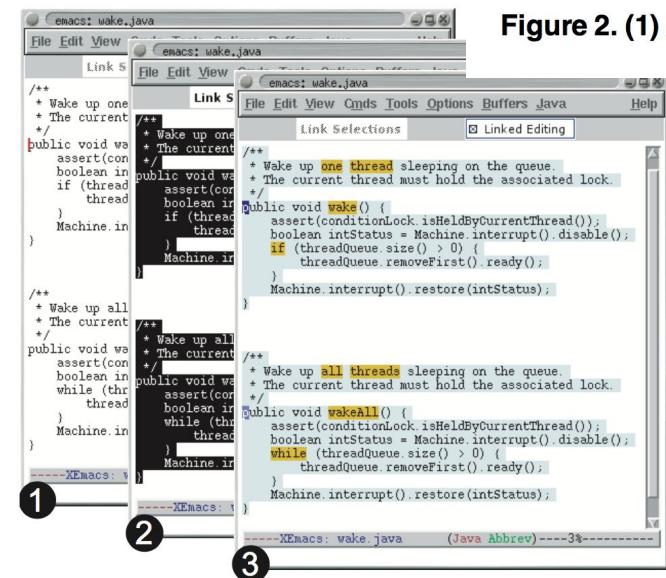


Figure 1. (1) Before linking two similar Java methods in Codelink. (2) After selecting the methods. (3) After linking the methods.

```
public void wakeAll() {
    assert(conditionLock.isHeldByCurrentThread());
    System.out.println("We're in wakeAll()");
    boolean intStatus = Machine.interrupt().disable();
    while (threadQueue.size() > 0) {
        ...
    }
}

public void wake() {
    ...
}
```

Figure 3. An elided clone looks similar to a function definition and use

## Code portals 2016

```
6
7
8 factorial = n => if n == 0
9     then 1
10    else n * factorial (n - 1);
11
12 main = if 7 == 0
13     then 1
14     else 7 * factorial (7 - 1);
```

# versions & diffs

Merge pull request #3866 from sebmarkbage/coreapi  
Reorganize Src Files for Isomorphic React Package  
Merge pull request #3749 from chrisgrovers/patch-1  
Fixed Formatting.  
Merge pull request #3874 from bhamodhi/patch-1  
Update CONTRIBUTING.md  
Merge pull request #3883 from ef718/docs-grammar  
Add documentation  
Followup to #3852, use https  
Merge pull request #3882 from wall-s/get-html5-minlength-working  
Get HTML5 minlength working  
Merge pull request #3847 from bloodyowl?option-flatten-children  
<option> addes children flatten, & warn if invalid  
Merge pull request #3719 from spicys/vdn2  
Add more context to DOM nesting warning  
Merge pull request #4365 from spicys/tac-no!  
Don't thread index through traverseAllChildren  
Merge pull request #3872 from spicys/nested-num  
Split message for deep numeric key warning too  
Merge pull request #3869 from zpao/jsx-cleanup  
Use Babel to transform JS in docs, update other calls  
(docs) Move pre-compiled JS files  
Don't run non-existent transform tests  
Merge pull request #3871 from spicys/group  
Deprecate reactjs Google Group  
Merge pull request #3870 from spicys/gh-3865  
Clarify ReactElement prop validation message  
Merge pull request #3861 from spicys/nested-v  
Split out warning message for nested key warning  
Merge pull request #3864 from cyverbry/patch-1  
bower.json: remove moot 'version' field  
Merge pull request #3857 from spicys/dev-rckv  
Set\_reactChildKeysValidated in dev mode only  
Merge pull request #3856 from chaseadamsio/chaseadamsio-patch-1  
Change object type to objects  
Merge pull request #3834 from chicoyzzy/remove\_jslint  
remove\_jslint comments  
Merge pull request #3837 from davideat/transition-group-appear  
added animate initial mounting section to animation docs  
0.14.0-alpha1  
Fix esprima-fb dependency  
Merge pull request #3852 from thewarpaint/patch-1  
Add Wikipedia link to Cross-site scripting on "XSS attack" string  
Follow-up to follow-up to #3718  
Merge pull request #3835 from zpao/fix-keys-keys-keys-test  
Followup to #3758 so we actually test things  
Merge pull request #3831 from MadLittleMods/tab-size-unit-less

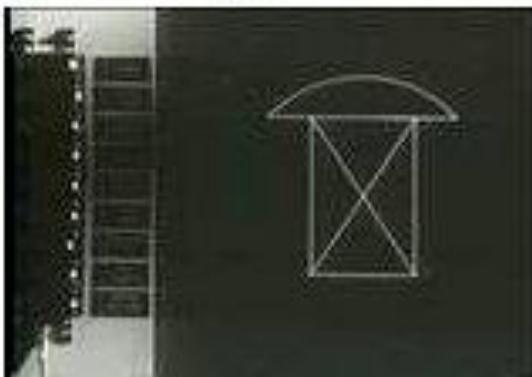
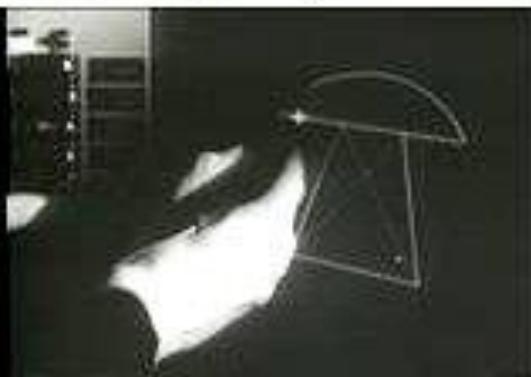
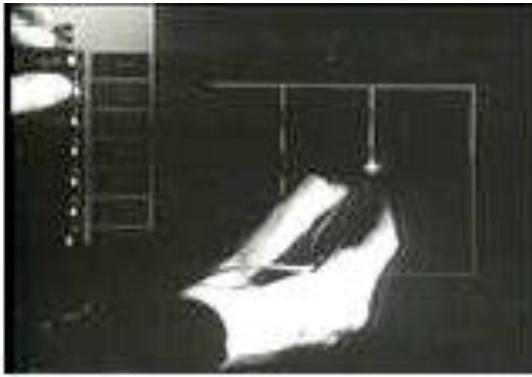
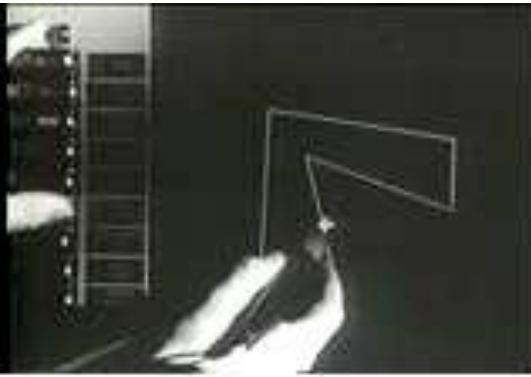
```
876     default:  
877       return undefined;  
878     }  
879   }  
880   /* visit slots and values top-down outside of derived values */  
881   visitStatic(visitor: (x: SlotOrValue) => void) {  
882     visitor(this);  
883     if (this.metadata) {  
884       assert(this.metadata.inSlot === this);  
885       assert(this.metadata.version === this.version);  
886       this.metadata.visitStatic(visitor);  
887     }  
888     if (this.derived) {  
889       // don't visit value to allow recursion  
890       if (this.source) {  
891         // bind source  
892         this.bind();  
893       }  
894     }  
895   }  
896   return;  
897 }  
898 assert(this.value.inSlot === this);  
899 assert(this.value.version === this.version);  
900 this.value.visitStatic(visitor);  
901 }  
902 equals(other: SlotOrValue): boolean {  
903   // FIXME: skip formulas?  
904   // FIXME: skip formulas?  
905   return other instanceof Slot && this.value.equals(other.value);  
906 }  
907   
908 default:  
909   return undefined;  
910 }  
911 }  
912 if (this.metadata) {  
913   assert(this.metadata.inSlot === this);  
914   assert(this.metadata.version === this.version);  
915   this.metadata.analyze();  
916 }  
917 if (this.derived) {  
918   if (this.source) {  
919     // bind source  
920     this.bind();  
921     // but don't instantiate ~  
922     return; ~  
923   }  
924   // instantiate calls and conditionals to analyse them ~  
925   this.value; ~  
926   return;  
927 }  
928 assert(this.value.inSlot === this);  
929 assert(this.value.version === this.version);  
930 this.value.analyze();  
931 }  
932 }  
933 equals(other: SlotOrValue): boolean {  
934   // FIXME: skip formulas?  
935   return other instanceof Slot && this.value.equals(other.value);  
936 }
```

```
877 1062 default:  
878 1063   return undefined;  
879 1064 }  
880 1065 }  
881 1066 }  
882 1067 /* visit slots and values top-down outside of derived values */  
883 1068 visitStatic(visitor: (x: SlotOrValue) => void) {  
884 1069 // analyze slots and values top-down outside of derived values */  
885 1070 analyze() {  
886 1071 visitor(this);  
887 1072 if (this.metadata) {  
888 1073 assert(this.metadata.inSlot === this);  
889 1074 assert(this.metadata.version === this.version);  
890 1075 this.metadata.visitStatic(visitor);  
891 1076 this.metadata.analyze();  
892 1077 }  
893 1078 if (this.derived) {  
894 1079 // don't visit value to allow recursion ~  
895 1080 if (this.source) {  
896 1081 // bind source  
897 1082 this.bind();  
898 1083 // but don't instantiate ~  
899 1084 return; ~  
900 1085 }  
901 1086 // instantiate calls and conditionals to analyse them ~  
902 1087 this.value; ~  
903 1088 return;  
904 1089 }  
905 1090 assert(this.value.inSlot === this);  
906 1091 assert(this.value.version === this.version);  
907 1092 this.value.visitStatic(visitor);  
908 1093 this.value.analyze();  
909 1094 }  
910 1095 }  
911 1096 equals(other: SlotOrValue): boolean {  
912 1097 // FIXME: skip formulas?  
913 1098 // FIXME: skip formulas?  
914 1099 return other instanceof Slot && this.value.equals(other.value);  
915 1100 }
```

```
872 switch (name) {  
873   case MetalD.prototypeField.name:  
874     let value = this.value;  
875     return value instanceof List ? value.prototypeField :  
876       undefined;  
877     default:  
878       return undefined;  
879     }  
880   }  
881   /* visit slots and values top-down outside of derived values */  
882   visitStatic(visitor: (x: SlotOrValue) => void) {  
883     visitor(this); ~  
884     if (this.metadata) {  
885       assert(this.metadata.inSlot === this);  
886       assert(this.metadata.version === this.version);  
887       this.metadata.visitStatic(visitor); ~  
888     }  
889     if (this.derived) {  
890       // don't visit value to allow recursion ~  
891       if (this.source) {  
892         // bind source  
893         this.bind();  
894       }  
895     }  
896     return;  
897   }  
898   assert(this.value.inSlot === this);  
899   assert(this.value.version === this.version);  
900   this.value.visitStatic(visitor); ~  
901 }  
902 equals(other: SlotOrValue): boolean {  
903   // FIXME: skip formulas?  
904   // FIXME: skip formulas?  
905   return other instanceof Slot && this.value.equals(other.value);  
906 }  
907   
908 default:  
909   return undefined;  
910 }  
911 }  
912 if (this.metadata) {  
913   assert(this.metadata.inSlot === this);  
914   assert(this.metadata.version === this.version);  
915   this.metadata.analyze(); ~  
916 }  
917 if (this.derived) {  
918   if (this.source) {  
919     // bind source  
920     this.bind();  
921     // but don't instantiate ~  
922     return; ~  
923   }  
924   // instantiate calls and conditionals to analyse them ~  
925   this.value; ~  
926   return;  
927 }  
928 assert(this.value.inSlot === this);  
929 assert(this.value.version === this.version);  
930 this.value.analyze(); ~  
931 }  
932 }  
933 equals(other: SlotOrValue): boolean {  
934   // FIXME: skip formulas?  
935   return other instanceof Slot && this.value.equals(other.value);  
936 }
```

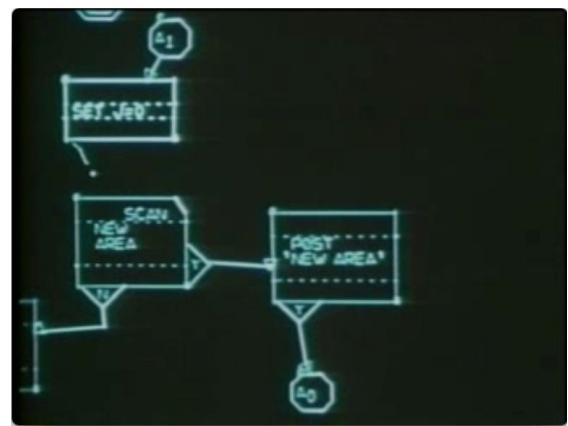
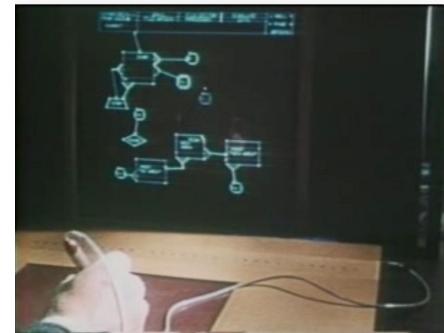
## Sketchpad

1963



## GRAIL

1969



# PYGMALION

1975

```
menu
icons
create
change
delete
copy
refresh
show
name
value
shape
body
opcodes
+
-
/
=
<
>
and
or
not
control
?
call
return
repeat
done
eval
others
remember
constant
define
display
draw
text
break
plot
exit
```

remembered  
smalltalk

mouse value

mouse

```
menu
icons
create
change
delete
copy
refresh
show
name
value
shape
body
opcodes
+
-
/
=
<
>
and
or
not
control
?
call
return
repeat
done
eval
others
remember
constant
define
display
draw
text
break
plot
exit
```

remembered  
smalltalk

mouse value

mouse

```
menu
icons
create
change
delete
copy
refresh
show
name
value
shape
body
opcodes
+
-
/
=
<
>
and
or
not
control
?
call
return
repeat
done
eval
others
remember
constant
define
display
draw
text
break
plot
exit
```

remembered  
smalltalk

mouse value

mouse

```
menu
icons
create
change
delete
copy
refresh
show
name
value
shape
body
opcodes
+
-
/
=
<
>
and
or
not
control
?
call
return
repeat
done
eval
others
remember
constant
define
display
draw
text
break
plot
exit
```

remembered  
smalltalk

mouse value

mouse

```
menu
icons
create
change
delete
copy
refresh
show
name
value
shape
body
opcodes
+
-
/
=
<
>
and
or
not
control
?
call
return
repeat
done
eval
others
remember
constant
define
display
draw
text
break
plot
exit
```

remembered  
smalltalk

mouse value

mouse

The argument to FACTORIAL is tested against the constant 1

6 ≠ 1

The "false" icon causes  $6 \times \text{FACTORIAL}(6 - 1)$  to be computed

Preparing for a recursive call on FACTORIAL

$\text{FACTORIAL}(5) = 120$  (The intermediate steps have not been shown.)

# Thinglab 1979

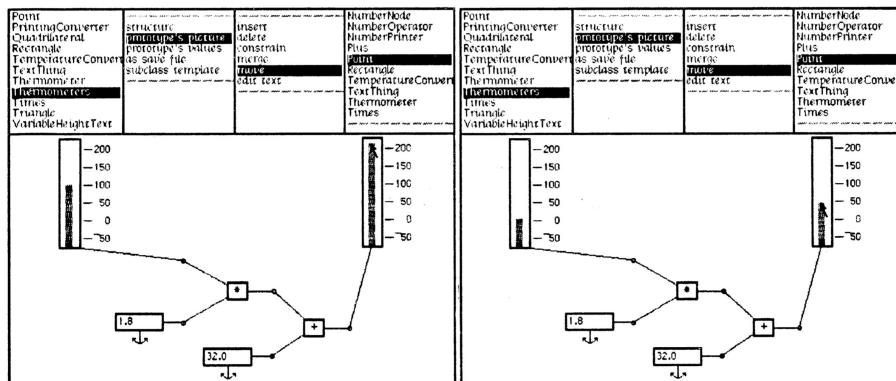


Figure 2.18 - The temperature converter with thermometers for input and output

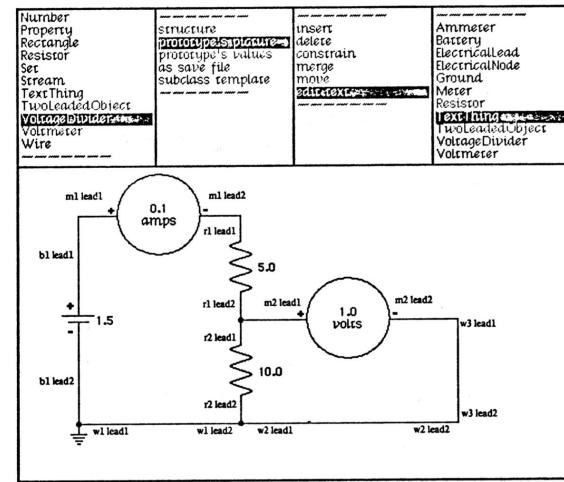


Figure 5.2 - A voltage divider

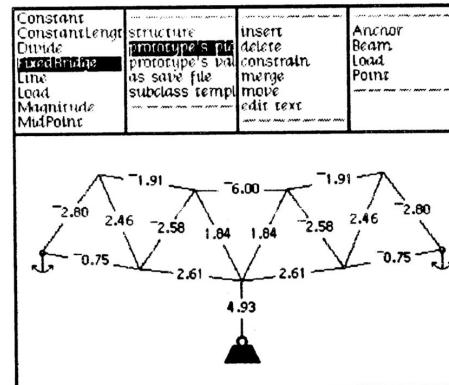


Figure 2.31 - A bridge under load

File Edit Eval Tools Windows Macros

## Tinker Examples

TYPEIN and EVAL

TYPEIN, but DON'T EVAL

NEW EXAMPLE for function

Give something a NAME

FILL in an ARGUMENT

EVALUATE something

UNFOLD something

COPY something

DELETE something

LEAVE Tinker

RETURN a value

Z

X Y

Table

Table

Tinker: (STACK (A BLOCK (NAME A)) (A

Predicate TRUE for: Result: "A on B", Code:

Result: #,(A BLOCK (:NAME A)), Code: FROM

Result: #,(A BLOCK (:NAME B)), Code: TO

Result: T, Code: (CLEARTOP? FROM)

Tinker Blocks World

A B

Table

Tinker: (STACK (A BLOCK (NAME X)) (A

Predicate FALSE for: Result: "X on Y", Code:

Result: #,(A BLOCK (:NAME X)), Code: FROM

Result: #,(A BLOCK (:NAME Y)), Code: TO

Result: NIL, Code: (CLEARTOP? FROM)

Tinker Blocks World

Z

X Y

Table

# Tinker 1981

'Tinker Functions

```
(DEFUN STACK (FROM TO)
  (MOVE-BLOCK FROM TO))
```

Tinker Blocks UI

```
(DEFUN STACK (FROM TO)
  (MOVE-BLOCK FROM TO))
```

A

B

Table

Z

X Y

Table

Tinker: (STACK (A BLOCK (NAME A)) (A

Predicate TRUE for: Result: "A on B", Code:

Result: #,(A BLOCK (:NAME A)), Code: FROM

Result: #,(A BLOCK (:NAME B)), Code: TO

Result: T, Code: (CLEARTOP? FROM)

Tinker Blocks World

A B

Table

'Tinker Functions

```
(DEFUN STACK
  (FROM TO)
  (IF
    (CLEARTOP? FROM)
    (MOVE-BLOCK FROM TO)
    (PROG-N
      (STACK (ABOVE FROM) THE-TABLE)
      (STACK FROM TO))))
```

Z

X Y

Table

Tinker: (HISTORY)

Defining (HISTORY):

Result: "A on B", Code: (STACK BLOCK-A BLOCK-B)

Result: "X on Y", Code: (STACK BLOCK-X BLOCK-Y)

Result: "D on E", Code: (STACK BLOCK-D BLOCK-E)

Tinker Blocks World

Tinker Block

H

G

F

D

E

E

H

G

F

Table

Table

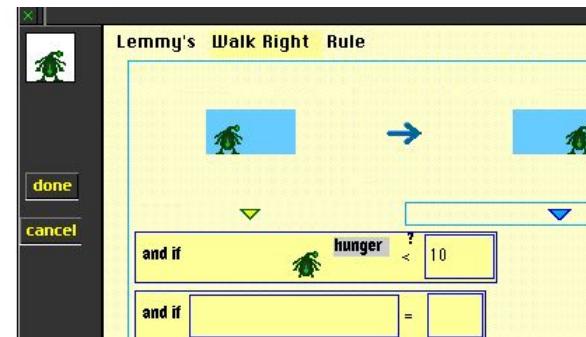
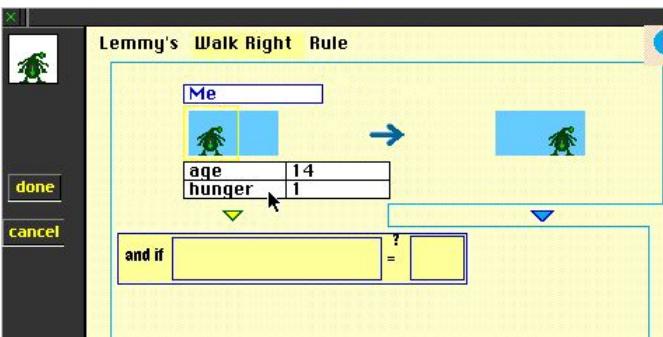
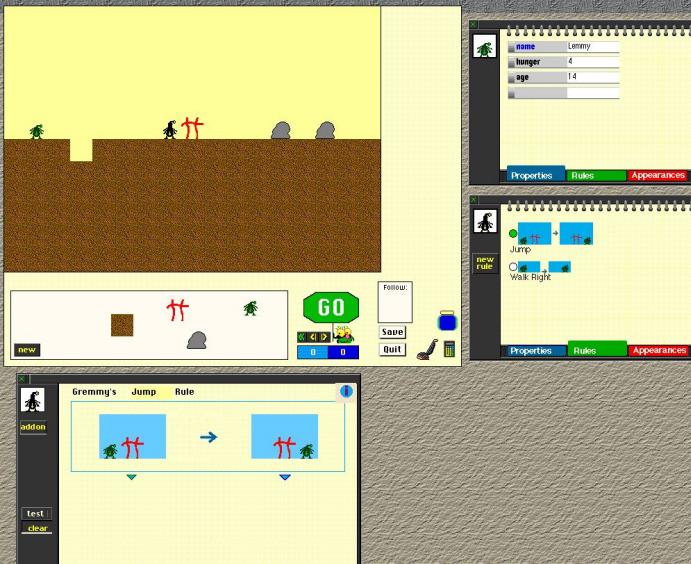
'Tinker Functions

```
(DEFUN STACK
  (FROM TO)
  (IF
    (CLEARTOP? FROM)
    (MOVE-BLOCK FROM TO)
    (PROG-N
      (STACK (ABOVE FROM) THE-TABLE)
      (STACK FROM TO))))
```

CL-USER| Eval... done.

# KidSim

## 1995



# ToonTalk 1996



Figure 5 -- A bird delivering a box with a nest to its nest

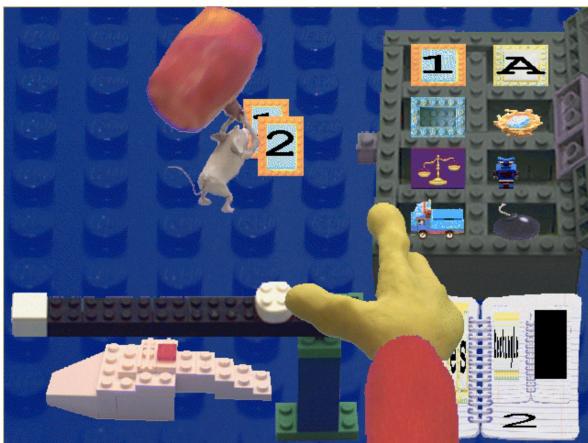


Figure 6 -- A mouse performing 1+2

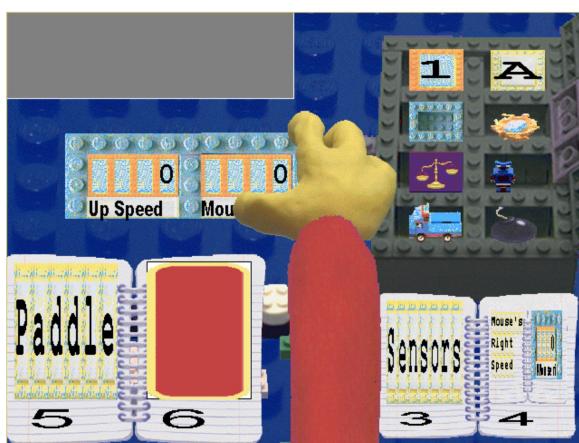


Figure 11 -- Just after building a cubby for controlling a paddle

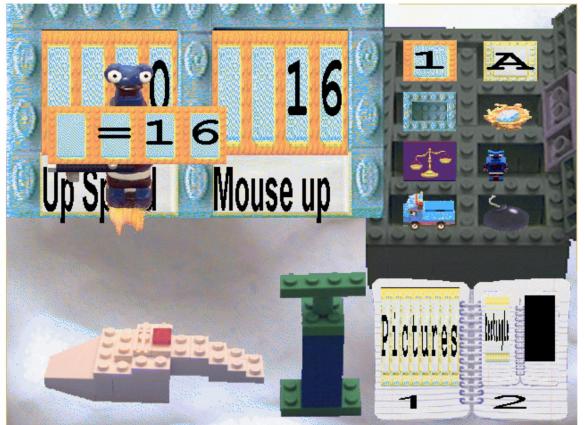


Figure 12 -- Training the robot so the paddle follows vertical mouse movements

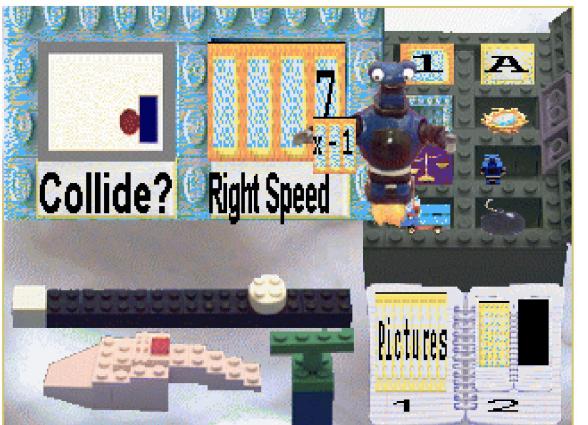
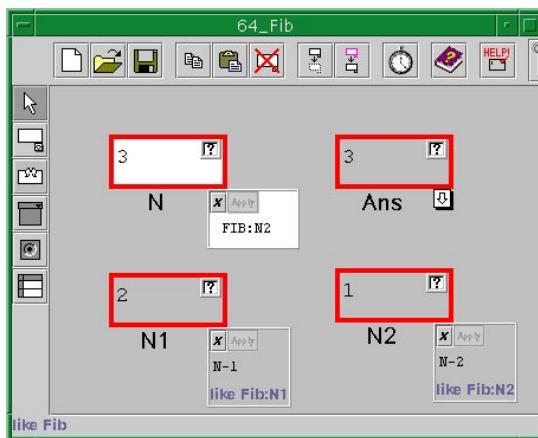
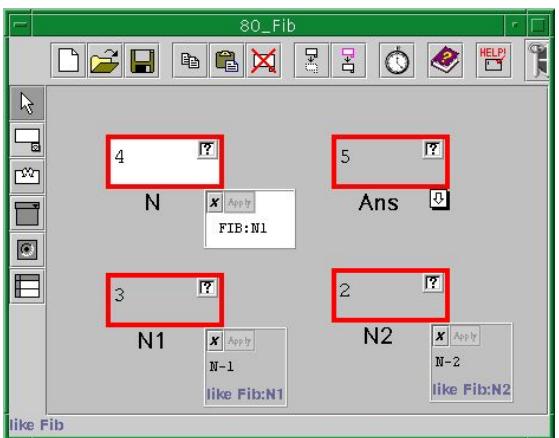
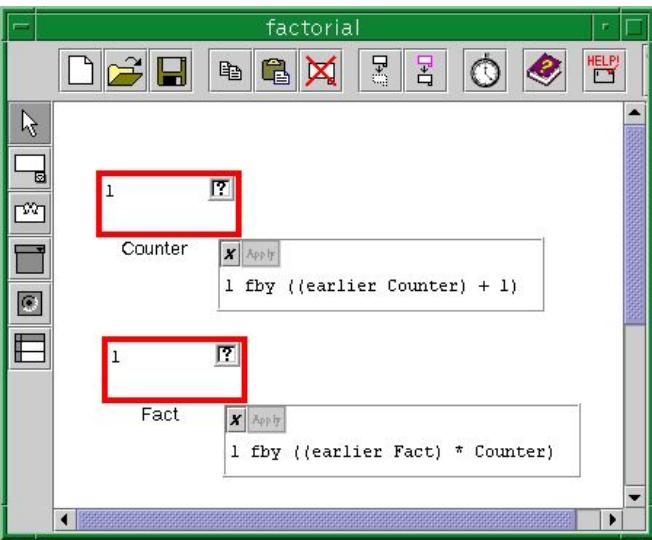
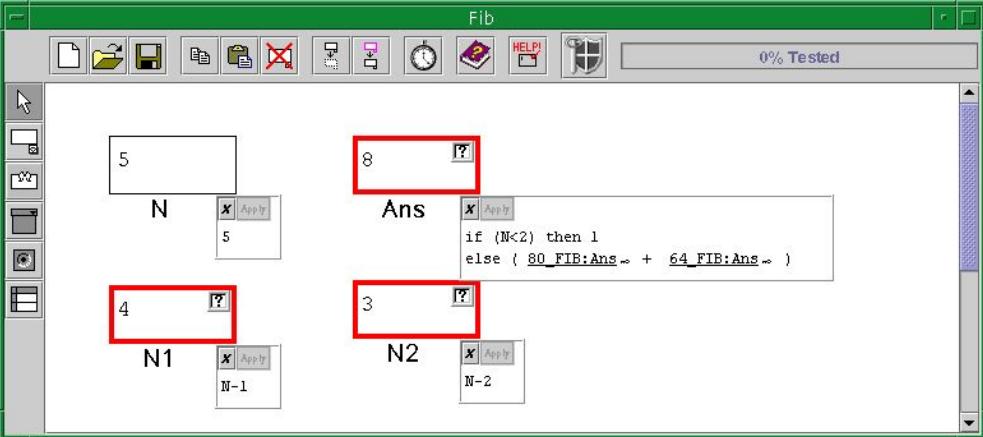


Figure 13 -- Training robot for ball bouncing



Figure 14 -- Adding another ball to the paddle game

# Forms/3 2001



# FAR 2001

The screenshot shows a window titled "End-user service programming with E-Speak". The main area displays a "White Flower Farm" web page. A query form is present, showing a search for flowers with a price less than 25. A "Formula for picture" dialog is open, containing a "Use Query" tab and a "Image table(2,7)" tab. Below the query form, a message says "We recommend the : tulip". A detailed description of tulips follows, along with a small image of tulips and a musical note icon. At the bottom, there is a table with columns: Flower, Color, Grows, Growin..., Price, Thumb..., Picture..., Picture..., and Thumb... (with a link to "Thumbnail"). The table contains rows for tulip and pansy. At the very bottom, there are tabs for "Database" and "Rules".

Figure 1: Snapshot of the flower advisor e-service in FAR as it is being created by the gardener. Everything shown in the web page area is a cell or table of cells, including the blocks of text, images, and so on.

Two screenshots of the "White Flower Farm" web page are shown. Both screenshots include a "Formula for picture" dialog with the "Image table(2,7)" tab selected. In the top screenshot, a "value" input field is set to 25, and the "qty" input field is set to 0.0. The bottom screenshot shows the "qty" field has been changed to 1.0. Below the dialog, a "Rules" section is visible. In the top screenshot, the "whenver" condition is "Always". In the bottom screenshot, the "whenver" condition is "(qty > 0)".

Figure 2: The flower advice example has been expanded, and the separator bar has been dragged upwards to make more room for the rules to be visible. (a) The gardener has added cells allowing a customer to purchase the recommended flowers if desired. The predicate of all these cells is initially "always". (b) The gardener changes the predicate, and the results are immediately reflected in the web page section. Since *qty* is currently 0, the predicate is false, and the cells' values are currently "no-value" (displays as blank).

# PlayGo 2003

PlayGo - GWTMemoryGame/CreateMemoryGameModel.umlseq - PlayGo

File Edit Diagram Navigate Search Project LSCEditor Run Window Help

Tahoma 9 B I A 100% LSC C 90%

LSC \*CreateMemoryGameModel.umlseq

LSC MemoryGameMove

User → memoryCard1 → memoryCard2 → memoryPanel

1 flipUp()

2 flipUp()

[0] (2,2,2,2) Cold

3 beep()

4 [0] (2,2,2,2) Hot

5 flipDown()

6 flipDown()

Outline System... Internal... Cross Re... Plug-ins

http://localhost:8888/GWTMemoryGame.html

Memory Game

Beep

```
sequenceDiagram
    User->>memoryCard1: flipUp()
    activate memoryCard1
    memoryCard1->>memoryCard2: flipUp()
    activate memoryCard2
    memoryCard2->>memoryPanel: flipUp()
    activate memoryPanel
    memoryPanel->>User: beep()
    deactivate memoryPanel
    deactivate memoryCard2
    deactivate memoryCard1
    User->>memoryCard1: flipDown()
    activate memoryCard1
    memoryCard1->>memoryCard2: flipDown()
    activate memoryCard2
    memoryCard2->>User: flipDown()
    deactivate memoryCard2
    deactivate memoryCard1
```

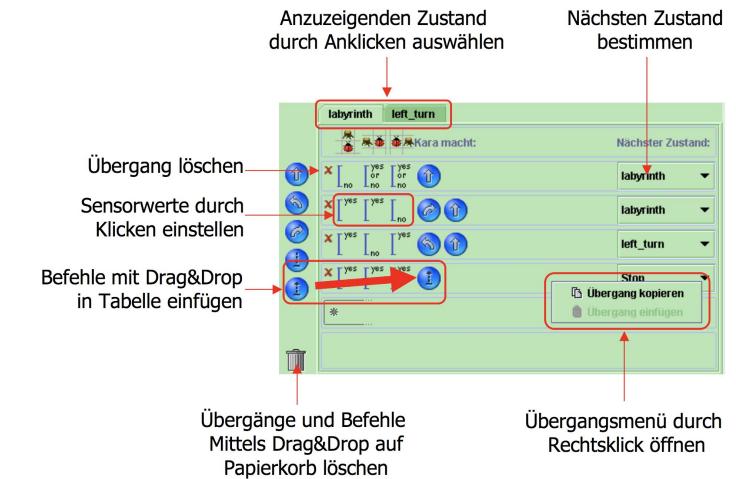
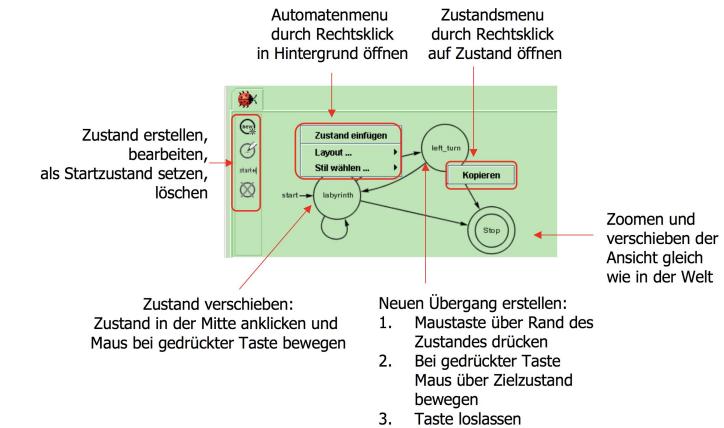
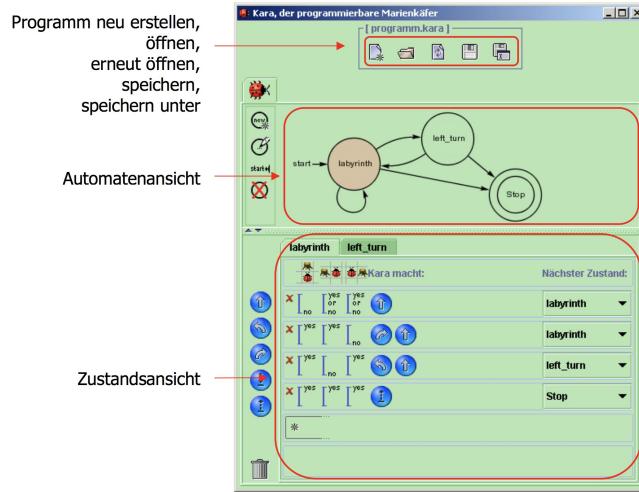
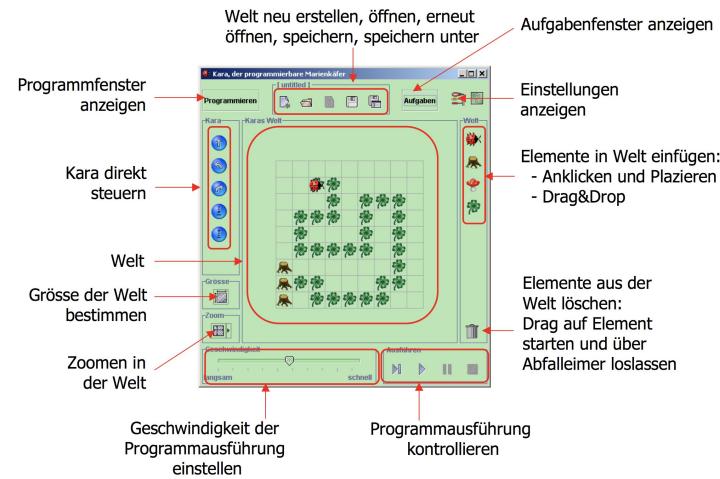
Play-In Play-Out NL Play-In Console Error Log Properties Problems Debug Development Mode

LSC Trace

- MemoryGameMove [0] (2,1,1,0) Cold
- MemoryGameMove [0] (2,2,2,1) Hot
- MemoryGameMove [1] (1,1,0,0) Cold
- MemoryGameMove [0] (2,2,2,2) Cold

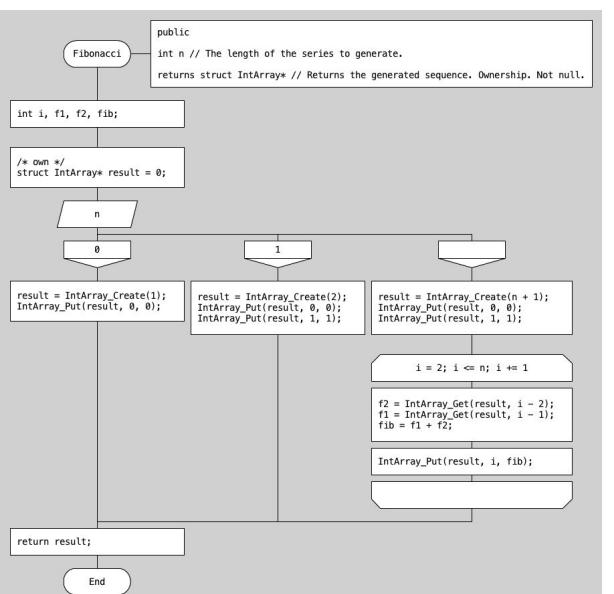
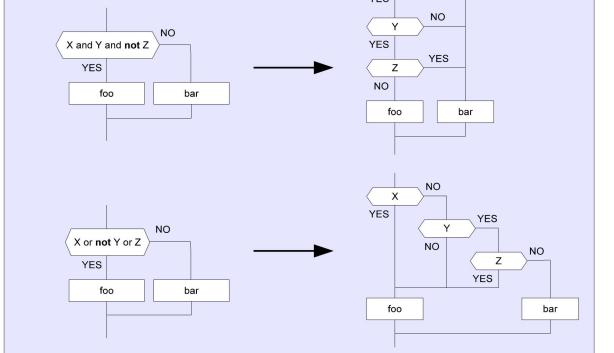
Reset To Current Cut

# Kara 2001



Text form is shorter but it makes the reader do more mental work.

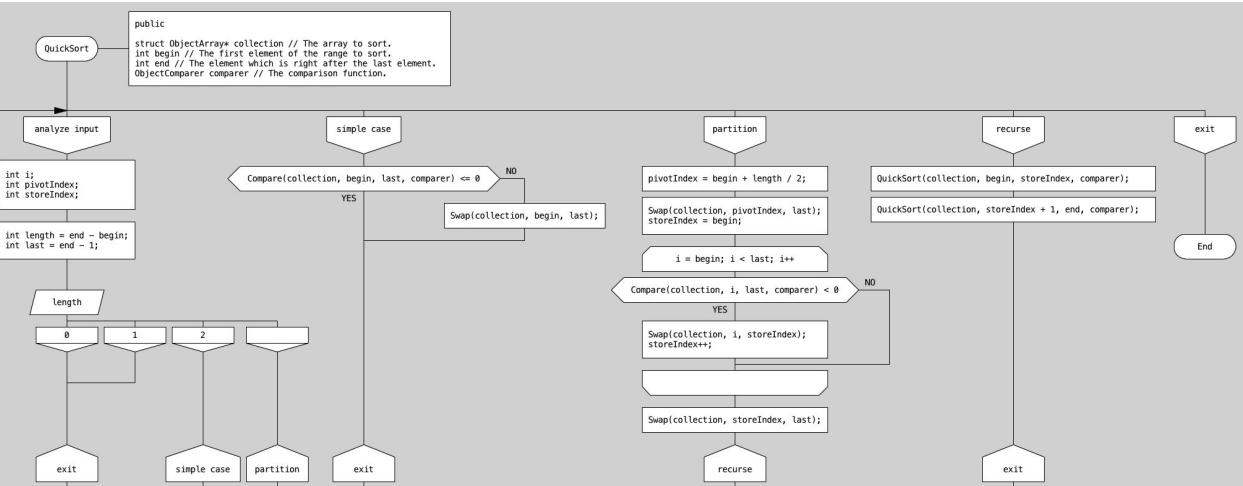
The reader has to imagine all possible outcomes of a logical expression each time he reads the diagram.



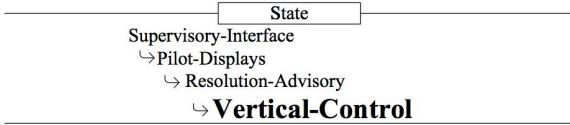
# Drakon

## 2011

Icon	Name of Icon	Icon	Name of Icon	Macroicon	Name of Macroicon
1	Title	14	Output	1	Title with parameters
2	End	15	Input	2	Fork
3	Action	16	Pause	3	Switch (number of cases N = 2)
4	Question	17	Period	4	SIMPLE loop by timer
5	Choice	18	Start timer	5	SWITCH loop by timer
6	Case	19	Synchronizer	6	FOR loop by timer
7	Headline	20	Realtime parallel process	7	WAIT loop by timer
8	Address	21	Comment	8	Insertion by timer
9	Insertion	22	Right comment	9	Output by timer
10	Formal parameters	23	Left comment	10	Start timer by timer
11		24	Loop arrow	11	Parallel process by timer
12	Begin of FOR loop	25	Silhouette arrow	12	Tree loop
13	End of FOR loop	26	Connector	13	
14		27	Concurrent process	14	



# SpecTRM-ML 1999



## DEFINITION

= Blank  
INITIALLY

= Other

Some RA-Strength <sub>s-277</sub>	in state Increase-2500fpm
Some Reversal <sub>s-282</sub>	in state Reversed
Composite-RA <sub>s-266</sub>	in state Climb
Composite-RA <sub>s-266</sub>	in state Descend
Corrective-Climb <sub>s-263</sub>	in state Yes
Corrective-Descend <sub>s-264</sub>	in state Yes
Crossing-Geometry <sub>m-388</sub>	

F	F	F
F	F	F
F	•	•
F	•	•
•	F	•
•	•	F
F	F	F

= Increase

Some RA-Strength <sub>s-277</sub>	in state Increase-2500fpm
Climb-Inhibit <sub>s-243</sub>	in mode Inhibited
Descend-Inhibit <sub>s-245</sub>	in mode Inhibited

T
T
T

= Crossing

Some Reversal <sub>s-282</sub>	in state Reversed
Composite-RA <sub>s-266</sub>	in state Climb
Composite-RA <sub>s-266</sub>	in state Descend
Some RA-Strength <sub>s-277</sub>	in state Increase-2500fpm
Corrective-Climb <sub>s-263</sub>	in state Yes
Corrective-Descend <sub>s-264</sub>	in state Yes
Crossing-Geometry <sub>m-388</sub>	

F	F	F	F
T	T	•	•
•	•	T	T
F	F	F	F
F	•	F	•
•	F	•	F
F	F	F	F

# Parnas Tables 2001

Procedure signature

Precondition

Vector Table

NewDirection (dir, loc, Req)			
$R_1 = (\text{bottom} \leq \text{loc} \leq \text{top}) \Rightarrow$			
value before	!ReqAbove(loc)! = !ReqBelow(loc)!	$\neg \text{!ReqAbove(loc)}! \wedge \neg \text{!ReqBelow(loc)}!$	!ReqAbove(loc)! $\wedge \neg \text{!ReqBelow(loc)}!$
value after	dir' = 'dir	down	up

value meets  
constraint

macros:  $\text{!ReqAbove!} = \exists f . \text{loc} < f \leq \text{top} \mid \text{Req}[f]$   
 $\text{!ReqBelow!} = \exists f . \text{Bottom} \leq f < \text{loc} \mid \text{Req}[f]$

macro NoChange:  
 $(\text{Req}'=\text{Req}) \mid (\text{loc}'=\text{loc})$

# Tabular Expressions 2011

$$f(x, y) \stackrel{\text{df}}{=} \begin{cases} x + y & \text{if } x > 1 \wedge y < 0 \\ x - y & \text{if } x \leq 1 \wedge y < 0 \\ x & \text{if } x > 1 \wedge y = 0 \\ xy & \text{if } x \leq 1 \wedge y = 0 \\ y & \text{if } x > 1 \wedge y > 0 \\ x/y & \text{if } x \leq 1 \wedge y > 0 \end{cases}$$

Inputs	x,y	Expression Name	f
x<=1			x>1
y<0			x-y
y==0			x+y
y>0			x*y
		x/y	y

# TouchDevelop 2011



dismiss

Search...



all APIs

display

```
boxed {  
    add  
    var i := 0  
    + var columns := math → floor(box → page width / 12)  
    for (var j < math → floor(♦tools → count / columns) ) {  
        boxed {  
            box → use horizontal layout  
            box → set margins(0, 0, 0, 1)  
            for (var k < columns) {  
                var tool := ♦tools → at(j * columns + k)  
                boxed {  
                    box → set width(11)
```

paste

copy

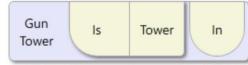
cut



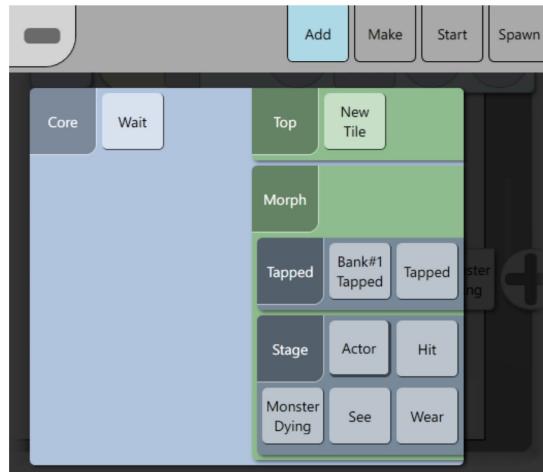
tools returns Collection of tool -- access the Collection of tool field

extract to var into local	extend select more	replace all in function abstract over	extract to parameter	strip 'for'		backspace
					more	undo
123 "...", not, true	(	)	,	async	<	>

# YinYang 2011

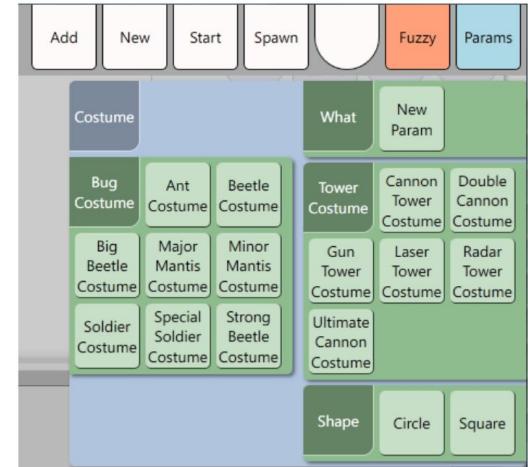


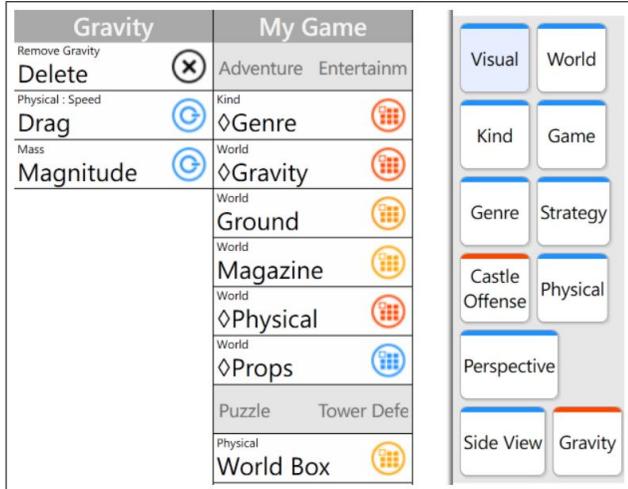
**Figure 4.** Definitions of the Attack, Tower, and Gun Tower tiles.



**Figure 7.** The root context menu (unscaled) for adding a new act in a Monster tile that extends the Actor tile. When the menu is active, programming content is grayed out. The menu has two parts: a top-positioned bar of menu modes, actions, and menu-wide options, and a menu of options that appears close to where the user tapped. The top level of Core panel of the menu contains the second-level Top and Morph panels; the second-level Morph panel contains the third-level Tapped and Stage panels.

the costume. However, since the Monster tile has not yet extended the Bug tile, bug costumes are not visible in this context menu. By selecting the Fuzzy option, additional costumes become visible in the menu:





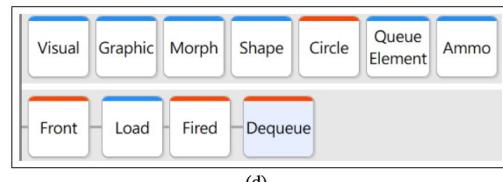
**Figure 6.** The YinYang editor after `Gravity` has been selected for the `My Game` object; code input menus are on the left; `My Game` labels the object’s root menu; `Gravity` labels the `Gravity` sub-menu; and traits extended by the `My Game` object are on the right.

Hamster***	
Contact	Damage
Queue	Dequeue
Deque	Front
Detect	Fade
Graphic	Fill
Fill	Dequeue
Fired	Freeze
Queue	Front
Morph	Deque
Front	Freeze
Gaming	Fired
Freeze	Game
Visual	Glow
Glow	Gestures
Visual	Gestures
Glow	Leave
Sling Shot	Glow
Load	Leave
Visual	Loop
Loop	Make
World	Morph
Morph	Movable
World	Morph
Morph	Movable
Movable	Score
Movable	Sensory
Graphic	Visual

(a) `Front`

(b) `Fired`

(c) `Dequeue`



**Figure 7.** Input menus for specifying behaviors of a `Hamster` object where (a) the object’s code is empty; (b) after the `Front` state is selected in (a); (c) after the `Fired` event is selected in (b); and (d) the object’s code after `Dequeue` is selected from (c).

# Sean McDermid APX

```
var p := ?  
val d := ?  
val unit := (50,50)  
var v := (50, 0)  
val f := (0, 50)  
val rr := (d,d) ÷ 2  
circle(p-rr, d, ?).draw(?, ?, ?)  
on tick:  
  val p1 = p + (v ÷ 50)  
  p = p1.clamp(?, ?)  
  v = (v) - (0, 0)
```

```
def inflect  
  (a : Number : exact, min, max):  
    get  
    val a0 = inflect_1
```

```
var p := ?  
val d := ?  
val unit := (50,50)  
var v := (50, 0)  
val f := (0, 50)  
val rr := (d,d) ÷ 2  
circle(p-rr, d, ?).draw(?, ?, ?)  
on tick:  
  val p1 = p + (v ÷ 50)  
  p = p1.clamp(rr, ?)  
  v = (v) + (f ÷ 50)
```

```
def inflect_0  
  (a : Number : exact, min, max):  
    zero  
    val a0 = a < min
```

item <sub>4</sub>	A
lerp	Num
max	Num
min	Num
one	ComparableNum
p	
p <sub>1</sub>	
point	
positionMouse	
rr	
screenSize	
step	Int
sw	Any
sx	Any
sy	Any
sz	Any
unit	
v	
zero	ComparableNum
zero	ComparableNum

# Sean-boxing 2017

```
var cx = document.querySelector(canvas).getContext(2d)
var total = results.reduce(sum choice) =>
    return sum + choice.count
0
this.helloNurse this this
1 + 2 + 3 + 4 + 5 + 1 + 1 + 2

a multi-line comment
that keeps going until here
```

```
var currentAngle = -0.5 * Math.PI;
before comment 1 + 1 after
var centerX = 300, centerY = 150
Add code to draw the slice labels in this loop.
results.forEach(result) =>
    var sliceAngle = result.count / total * 2 * Math.PI
    cx.beginPath()
    do comments show up here?
    cx.arc(centerX | centerY | 100 | currentAngle
        currentAngle + sliceAngle
    currentAngle += sliceAngle
    cx.lineTo(centerX | centerY)
    cx.fillStyle = result.color
    cx.fill()
```

# Greenfoot 2006

*This is what the rabbit does most of the time - it runs around. Sometimes it will breed or die of old age.*

```
public void act(List newRabbits)
```

```
if (isAlive())
    giveBirth(newRabbits)
    var Location newLocation
    newLocation = getField().freeAdjacentLocation(getX(), getY())
    if (newLocation != null)
        setLocation(newLocation)
    else
        setDead()
```

*Increase the age.*

*This could result in the rabbit's death (of old age).*

```
public void incrementAge()
```

```
setAge(getAge() + 1)
if (getAge() > MAX_AGE)
    setDead()
```

*Describe your method here...*

```
public void act()      overrides method in Actor
```

```
var String s
int x
```

```
while (x < 42)
    System.out.println("x=" + x)
    s = Scanner.readLine()
    if (s == null)
        break
    x = x + 1
```

**Figure 4. Code representation in a frame-based editor**

**Figure 6. Representation of a break statement**



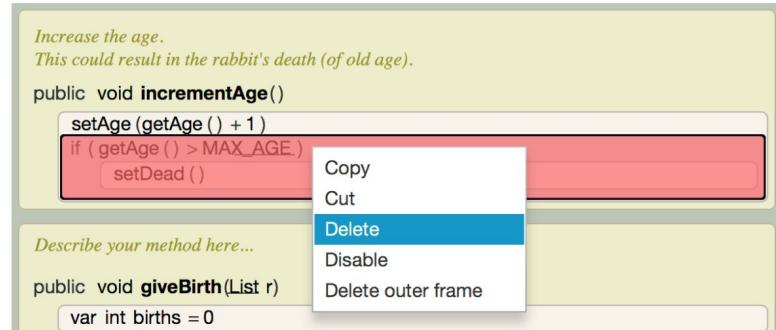
**Figure 7. An if-statement with empty slots**



**Figure 8. Optional text slots: invisible without focus (left) and visible when holding keyboard focus (right)**



**Figure 9. Cursors: a frame cursor (left) and a text cursor (right)**



**Figure 10. Preview of delete operations**

Increase the age.  
This could result in the rabbit's death (of old age).

```
public void incrementAge()  
    setAge (getAge () + 1)  
    if ( getAge () > MAX_AGE )  
        setDead ()
```

Figure 11. A disabled frame

Describe your method here...

```
public type name(paramType paramName, paramType paramName)
```

Figure 13. A method frame with empty slots

Describe your method here...

```
private type name()  
private  
protected
```

Figure 14. Text entry in a choice slot

Describe your method here...

public void act()     overrides method in Actor

```
if ( getAge () > MAX_AGE )  
    setDead ()
```

Increase ...

This could result in the rabbit's death (of old age).

public void incrementAge()

```
setAge (getAge () + 1)  
if ( getAge () > MAX_AGE )  
    setDead ()
```

Describe your method here...

public void act()     overrides method in Actor

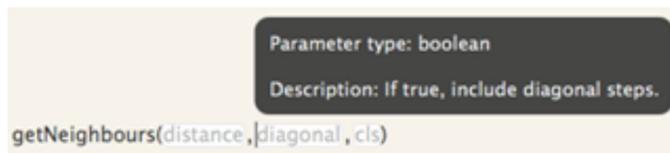
```
if ( getAge () > MAX_AGE )  
    setDead ()
```

Increase ...  
This

public void incrementAge()

```
setAge (getAge () + 1)  
if ( getAge () > MAX_AGE )  
    setDead ()
```

Figure 12. Cursor indicates valid (top) and invalid (bottom) drop targets. The drag source (at bottom of each) is blurred during drag.



**Figure 15. Prompt text and tooltip for parameters**

```
class Animal extends Actor
Imports ▶
Fields
    var private int age
    private boolean alive
Constructors
    public Animal ()
Methods
    public boolean isAlive ()
    public void setDead ()
    public int getAge ()
    public void setAge (int age)
    public Field getField ()
```

```
Ball ×
public void process(int ratio)
read = read(buffer)

height = getWorld().getHeight()
x = Greenfoot.getRandomNumber(height)
y = Greenfoot.getRandomNumber(300)

if (ratio > 0)
    while (read == 0 && a < 5)
        if (Math.hypot(x - getX(), y - getY()) > 20)
            setLocation(x, y)
            teleported = true

        setImage("crab.png")
        a = a + 1

        add(42)
        turn(5)

    else
        sb.append(" (-) negative")
        if (a == 5)
            System.out.println("read attempt failed")

        System.out.print("should not be negative")
```

**Figure 17. The Bird's Eye View**

**Figure 16. Method signature pinned to top of screen**

### class Rabbit extends Animal ▾

```
public World getWorld()
public void turnTowards(int x, int y)
protected boolean intersects(Actor other)
public int getRotation()
protected void addedToWorld(World world)
public void turn(int amount)
protected List<A> getObjectsInRange (int radius, Class<A> cls)
public void act()
protected boolean isTouching (Class<? extends Actor> cls)
public W extends World getWorldOfType (Class<W> worldClass)
protected A extends Actor getOneObjectAtOffset (int dx, int dy, Class<A> cls)
public void setLocation (int x, int y)
protected A extends Actor getOneIntersectingObject (Class<A> cls)
public void move (int distance)
public void setRotation (int rotation)
public boolean isAtEdge ()
protected List<A> getObjectsAtOffset (int dx, int dy, Class<A> cls)
protected List<A> getNeighbours (int distance, boolean diagonal, Class<A> cls)
public int getX()
```

**Figure 18. List of inherited methods**

*Act - do whatever the crab wants to do. This method is called whenever the 'Act' or 'Run' button gets pressed in the environment.*

public void **act()** overrides method in Actor

```
move (5)
checkKeyPress ()
lookForWorm ()
```

**Figure 19. The override annotation**

### turtle-crab - Stride

*is called whenever ent.*

*or*

*en pressed.*

*eight worms, we win.*

### Commands

<input type="checkbox"/> Call method	method-na
<input checked="" type="checkbox"/> Assignment	variable = n
<input type="checkbox"/> Variable declaration	var type na
<input type="checkbox"/> If	if ( condition )
<input type="checkbox"/> For-each loop	for each ( va
<input type="checkbox"/> While loop	while ( cond
<input type="checkbox"/> Return	return expre
<input type="checkbox"/> Comment	// Commer
<input type="checkbox"/> Blank	
<input type="checkbox"/> Switch	switch ( exp
<input type="checkbox"/> Try/catch	try
<input type="checkbox"/> Break	break
<input type="checkbox"/> Throw	throw expre

Cheat Sheet

**Figure 20. The “Cheat Sheet”, which appears on the right of the editor**

*Act - do whatever the crab wants to do. This method is called whenever the 'Act' or 'Run' button gets pressed in the environment.*

public void **act()**     overrides method in Actor

checkKeyPress()

move(5)

get()

Greenfoot.getImage()

List<Actor> getIntersectingObjects(Class<Actor>)

List<Actor> **getNeighbours(int, boolean, Class<Actor>)**

List<Actor> getObjectsAtOffset(int, int, Class<Actor>)

List<Actor> getObjectsInRange(int, Class<Actor>)

If it finds one intersecting object, return it

Actor getOneIntersectingObject(Class<Actor>)

If it finds one object at offset, return it

Actor getOneObjectAtOffset(int, int, Class<Actor>)

public int getRotation()

World getWorld()

World getWorldOfType(Class<World>)

int getX()

int getY()

Related:

String Greenfoot.getKey()

int Greenfoot.getMicLevel()

MouseInfo Greenfoot.getMouseInfo()

int Greenfoot.getRandomNumber(int)

void Greenfoot.setSpeed(int)

void Greenfoot.setWorld(World)

void setImage(String)

void setImage(GreenfootImage)

Showing common options. Press Ctrl+Space again to see all options

if (isTouching(Worm.class))

List<Actor>  
Class<Actor>

Return the  
This me  
of the in  
objects

All cells  
'distance'  
in the fo  
depend  
distance  
cells, (1

Param

distanc

Figure 21. Code completion in Stride

File Edit Text Project Browse Debug Windows Help

Project: tiles (inactive)

Contents of tiles (inactive)

```

Dyl-a
dylan-framework (inactive)
Dylan-user
  documentati
    Copyright (C) 1994, Apple Computer, Inc. All rights reserved.
    MacDylan Tiles Program
  Version 1
  Version 2
  Version 3
  Version 4
  To Do
library & module
  Copyright (C) 1994, Apple Computer, Inc. All rights reserved.
  define library tiles
    use Dylan;
    use mac-toolbox;
    use Dylan-Framework;

    export tiles;
  end library;

  define module tiles
    use Apple-Dylan;
    use dylan-framework,
      // exclude sealed generic with conflicting name
      // This is because the framework's <grid-view> is
      // a bit like our <cell-view>.
    exclude: { point-to-cell };
    use mac-toolbox;

    export
      init-tiles,
      run-tiles;
  end module;

  tiles
    cell view
      /* Copyright (C) 1994, Apple Computer, Inc. All rights reserved. */
      cells
        define class <cell> <object>
          // Cells sit in a view, and have (x,y) coordinates
          // and neighbors.
          slot view :: <view>;
          required-init-keyword: view::;

          slot row :: <integer>;
          required-init-keyword: row::;

          slot column :: <integer>;
          required-init-keyword: column::;

```

Dylan-user

# Dylan 1992

Function Family of + (object1, object2)

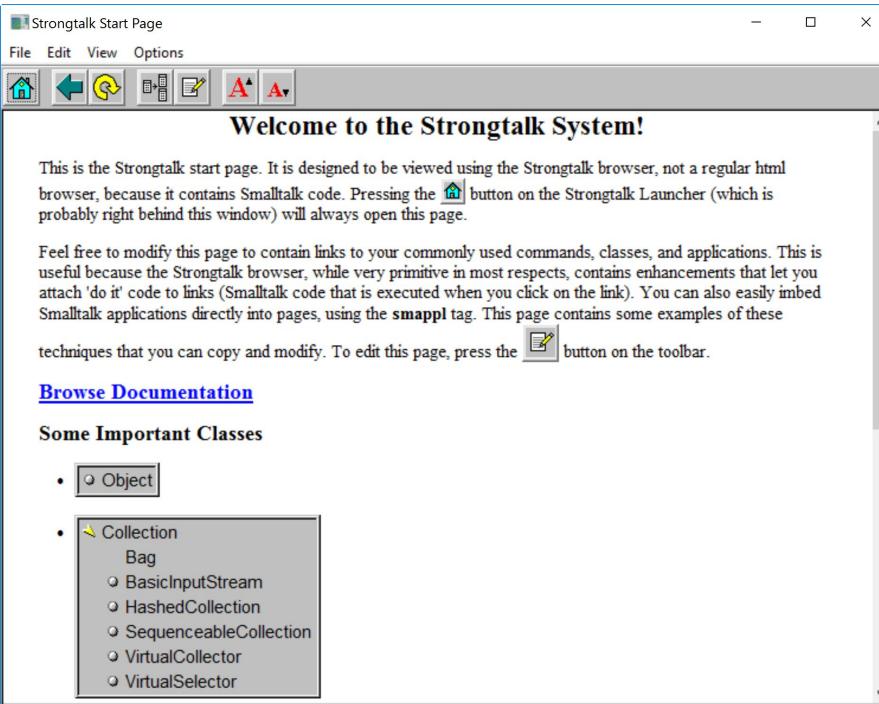
```

+ (x :: <small-integer>, y :: <small-integer>) => + :: <integer>
  ↗ No source code available
  + (p :: <machine-pointer>, o :: <integer>)
  + (o :: <integer>, p :: <machine-pointer>)
  + (object1, object2)
  + (r1 :: <ratio>, r2 :: <ratio>)
  + (r :: <ratio>, n :: <integer>)
  + (n :: <integer>, r :: <ratio>)
  + (x :: <big-integer>, y :: <small-integer>) => v :: <integer>
  + (x :: <small-integer>, y :: <big-integer>) => v :: <integer>
  + (x :: <big-integer>, y :: <big-integer>) => v :: <integer>
  + (x :: <double-float>, y :: <double-float>) => result :: <double-float>
  + (x :: <double-float>, y :: <rational>) => result :: <double-float>
  + (x :: <rational>, y :: <double-float>) => result :: <double-float>
  + (p1 :: <point>, p2 :: <point>) => sum :: <point>
  + (a :: <number>, b :: <point>) => result :: <point>
    define method \+ (a :: <number>, b :: <point>) => result :: <point>
      point(a+b.y, a+b.v);
    end method;
  + (a :: <point>, b :: <number>) => result :: <point>
    define method \+ (a :: <point>, b :: <number>) => result :: <point>
      point(a.b + b, a.v + b);
    end method;
  + (a :: <rect>, b :: <rect>) => result :: <rect>
  + (a :: <rect>, b :: <number>) => result :: <rect>
  + (a :: <number>, b :: <rect>) => result :: <rect>
  + (a :: <rect>, b :: <point>) => result :: <rect>

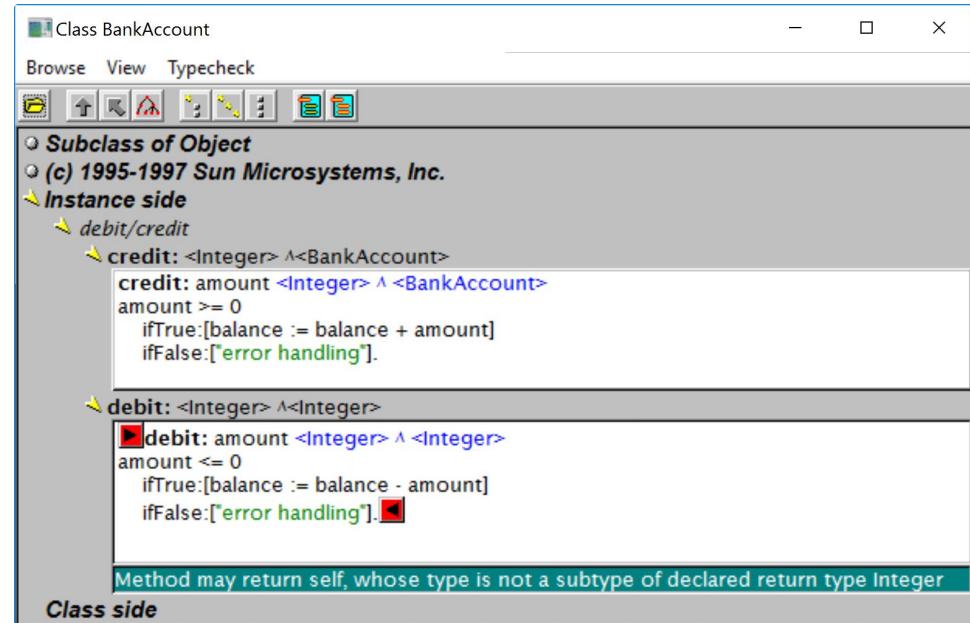
```

Dylan

# Strongtalk 1993

A screenshot of the Strongtalk Start Page window. The title bar says "Strongtalk Start Page". The menu bar includes "File", "Edit", "View", and "Options". The toolbar has icons for Home, Back, Forward, Stop, Refresh, and two A-shaped buttons. The main content area displays the text "Welcome to the Strongtalk System!". Below it, a paragraph explains the system's nature and how to modify pages. It also mentions the "smapp!l" tag and provides instructions for editing the page. A link "Browse Documentation" is present. A section titled "Some Important Classes" lists "Object" and "Collection" with its subclasses: "Bag", "BasicInputStream", "HashedCollection", "SequenceableCollection", "VirtualCollector", and "VirtualSelector".

This shows a literate programming style. In particular, we have two Strongtalk class hierarchy browsers embedded in a document. The second, for Collection and its subclasses, is partly expanded. The browsers are of course live and can be interacted with (e.g., expanded, collapsed, inspected etc.) from within the document.

A screenshot of the Class BankAccount browser window. The title bar says "Class BankAccount". The menu bar includes "Browse", "View", and "Typecheck". The toolbar has icons for Home, Back, Forward, Stop, Refresh, and three other icons. The main content area shows a tree view of methods for the "BankAccount" class. The "credit:" method is expanded, showing its implementation: "credit: amount <Integer> ^ <BankAccount> amount >= 0 ifTrue:[balance := balance + amount] ifFalse:[error handling].". The "debit:" method is also partially expanded, showing its implementation: "debit: amount <Integer> ^ <Integer> amount <= 0 ifTrue:[balance := balance - amount] ifFalse:[error handling].". A message at the bottom states "Method may return self, whose type is not a subtype of declared return type Integer".

The class browser differs from the classic Smalltalk one. It shows a tree view of the methods, so multiple methods can be viewed simultaneously. There are options to open all methods, which gives you something a bit like looking at a file.

Note the **credit:** method - the edges of the method editor give the visual impression of sticking out rather than being recessed, as they are in the **debit:** method. This is a cue that the text has been modified but not yet saved.

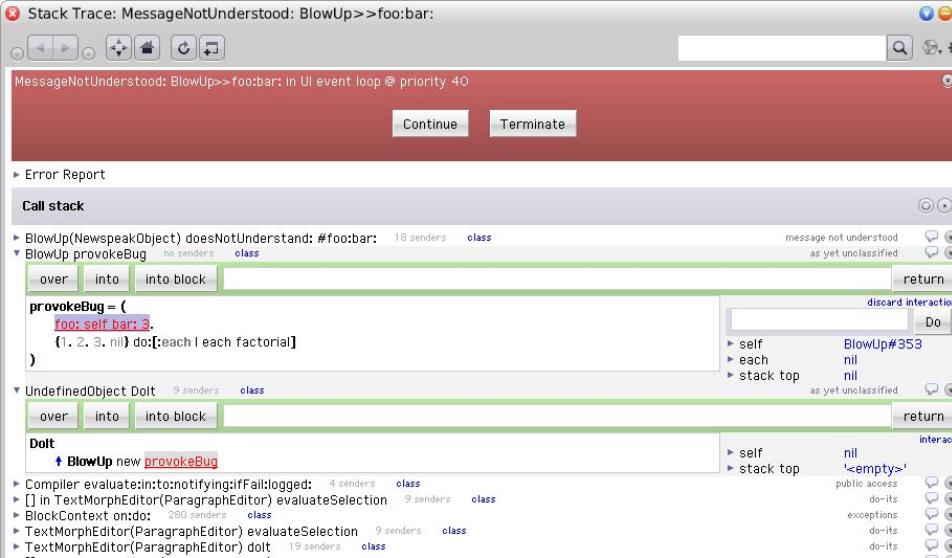
The **debit:** method has been typechecked. The type errors are reported, conventionally (though it was new back then) at the bottom of the method editor. The error is highlighted by two arrow widgets.

# Hopscotch 2008

Hopscotch is a navigation based UI, like a web browser or current mobile platforms. Hence, the window has some general purpose facilities at the top - a search pane, navigation arrows, history button etc. **There are no tabs.** The history page is more scalable. Navigation makes any other open view accessible within two clicks - history button and the desired view.

The class being viewed is shown below. It is a tree view, inheriting something from the Self and Strongtalk. We have a nested class, SequentialParser, expanded, and with it one of the methods is expanded in place. This has the same property the Strongtalk browser had, that you can view multiple methods at once.

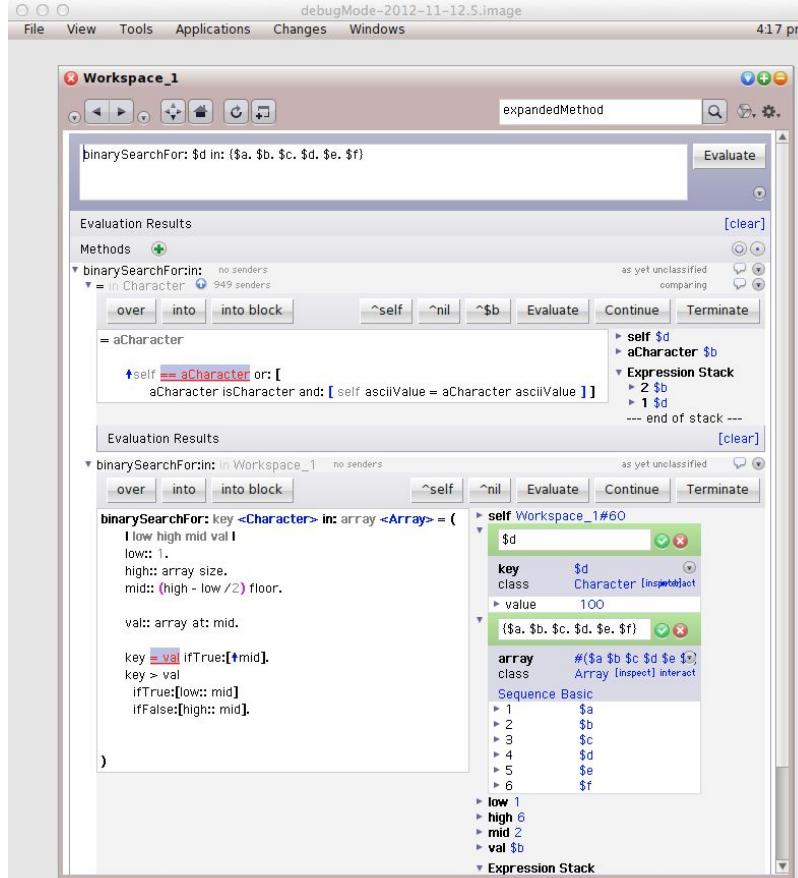




Newspeak-debugger. It inherits from the Self and Strongtalk debuggers, the stack view, which allows multiple frames to be open at once and corresponds to the natural, white board visualization of the call stack. I actually have a post with screenshots of those:

<https://gbracha.blogspot.com/2008/07/debugging-visual-metaphors.html>.

An open frame shows the method code on the left, and local and parameter bindings on the right. Each variable can be expanded into an inspector to see its fields, access its class, evaluate expressions within its scope etc.



Exemplar workspace. See also <https://gbracha.blogspot.com/2013/04/making-methods-live.html>.

The workspace has an editable pane at the top, in which multiple snippets can be kept and evaluated, much like the classic Smalltalk workspace (which is notably different than a REPL, precisely because multiple things can be kept around).

Below, the workspace has an area for workspace-specific methods. This is unusual. In fact, the workspace is an instance of a custom-made class, which is where the methods really go. The method is presented like an activation record in the debugger, so in addition to the code, as shown in the conventional class browser, we have live values, we can execute the code etc. In this shot, we are stepping into the call to =. Essentially the same debugger UI as above.

# JOSS

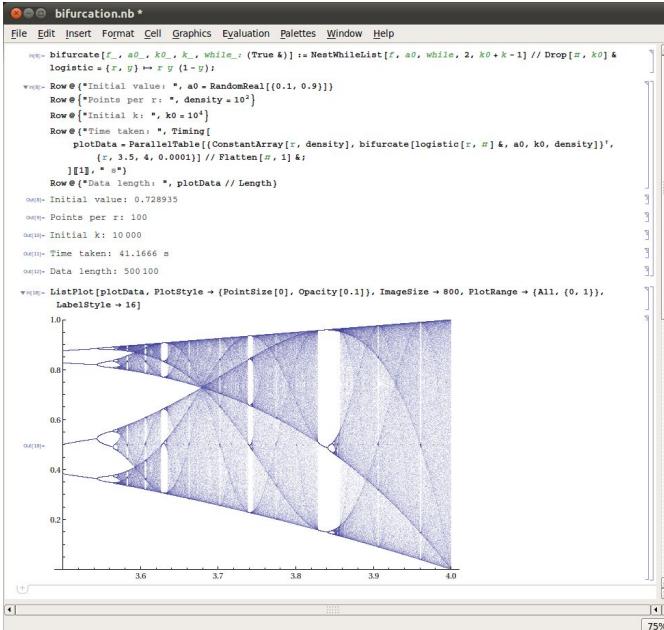
1966

```
1143 6/13/70 #44 ral 1 [2]
```

```
Type "yes" if 23923<45+62.
Type 45+62 =
2790
Type "yes" if 23923>45+62.
yes
Type"yes" if 97843>97843.
Eh?
SORRY.
Eh?
Let x = 56.75.
Type x.
x =
4200
Let y = x-75.
Type y.
y =
4125
Type x.y.
Eh?
Type x.y.
x.y =
1.7325*10^87
Type x.y.
x.y =
1.01818182
Type x.y.
x.y =
9829.52729
Let v=x*9986533/4.
Type v*x.y.
v*x.y =
4.3326893*10^13
Type v.
Eh?
Type v.
v =
2.50083075*10^96
Type x.
x =
4200
Type y.
y =
4125
Type v*x.y/7.
v*x.y/7 =
1.05034898*10^810
Type v/x.
v/x =
595.435893
Type v/x*y89/67*34.
v/x*y89/67*34 =
4765.60007
Type v*x.
v*x =
2.49663075*10^46
```

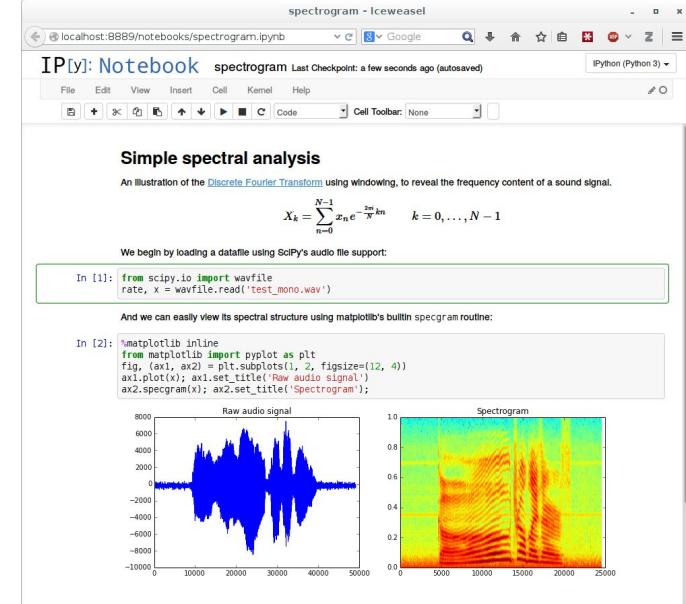
# Mathematica

1988



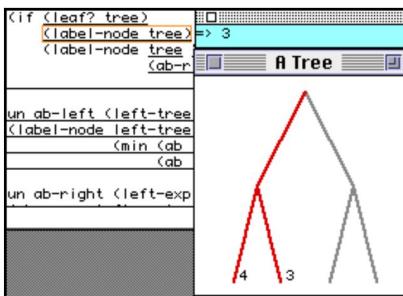
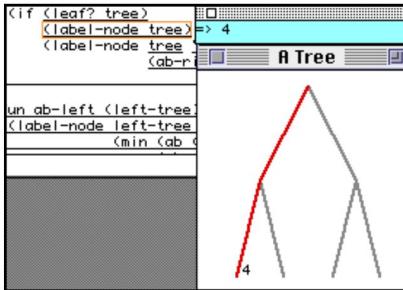
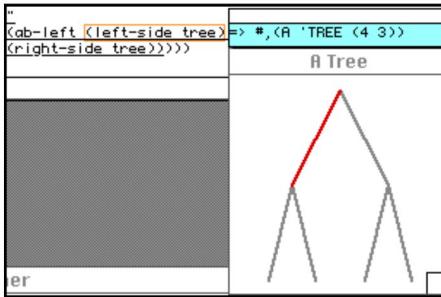
# IPython

2001

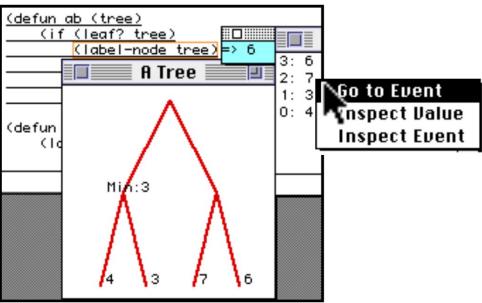


# ZStep 95

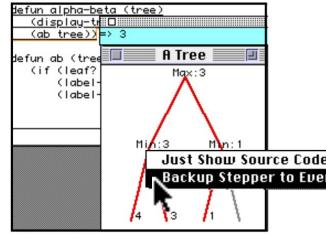
Go to end of program —————  
 Show value of expression, without stopping —————  
 Single step —————  
 Single step backwards —————  
 Back up from value to expression —————  
 Go to beginning of program —————  
 Single step "graphically" —————  
 Single step backwards "graphically" —————



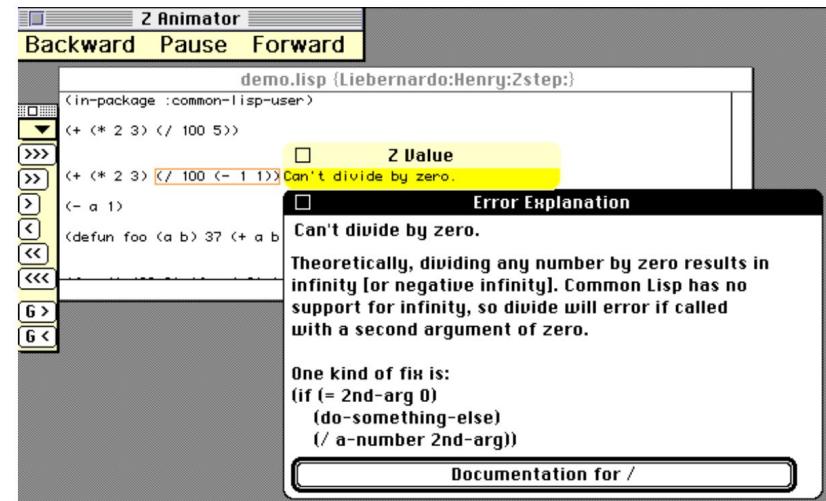
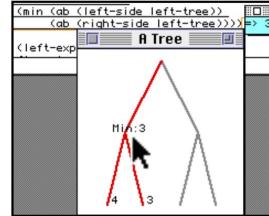
The value window moves through the code



Three successive "graphic steps"



Clicking on a graphical object backs up the stepper to the event which drew it



One kind of fix is:  
`(if (= 2nd-arg 0)  
 (do-something-else)  
 (/ a-number 2nd-arg))`

Documentation for /

The history of values of an expression

# Omniscient Debugger

2007

Omniscient Debugger 29.Dec.06 - com.lambda.Debugger.Demo

File Run Trace Filter Previous Event 532 [1273] Demo.java:198

Threads

```
<main_7>
<Sorter_0>      <Sorter_1>
<Sorter_1>      <Sorter_2>
<Sorter_2>      <Sorter_3>
<Sorter_3>
-- <Sorter_4> --
-- <Sorter_5> --
-- <Sorter_6> --
-- <Waiter_8> --
```

Stack

```
<DemoRunnable_3>.run()
<Demo_0>.sort(0, 5)
<Demo_0>.sort(3, 5)
<Demo_0>.average(3, 5)
```

Locals

* start	3
* end	5
* sum	0
* i	3

this

```
<Demo_0>
quick  <Demo_1>
c      'X' (88)
b      '=' (61)
array   int[20]_0
```

Method Traces

```
**<DemoRunnable_3>.run() -> void
<Demo_0>.sort(0, 5) -> void
  <Demo_0>.average(0, 5) -> 240
  DemoRunnable.new(<Demo_0>, 0, 2) -> <DemoRunnable>
  Thread.new(<DemoRunnable_6>, "Sorter") -> <Sorter_6>
  <Sorter_6>.start() -> void
  <Demo_0>.sort(3, 5) -> void
    <Demo_0>.average(3, 5) -> 483
    <Demo_0>.sort(3, 4) -> void
      <Demo_0>.sort(5, 5) -> void
        sort -> void
        <Sorter_6>.join() -> void
        sort -> void
      run -> void
```

Code

```
return;
}

public int average(int start, int end) {
    int sum = 0;
    for (int i = start; i < end; i++) {
        sum += array[i];
    }
}
```

TTY Output

```
-----ODB Demo Program-----
A badMethod threw: java.lang.NullPointerException
Starting QuickSort: 20
-- Done sorting --
-- 0 1 --
-- 1 0 --
-- 2 237 --
-- 3 243 --
```

Objects

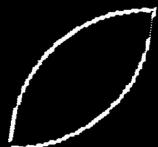
<Demo_0>	quick	<Demo_1>
c	'X'	(88)
b	'=	(61)
array	int[20]_0	
*	19	1968
*	18	1962
	17	1725
	16	1719
*	15	1476
*	14	1470
	13	1221
	12	1233
	11	1227
*	10	984
	9	978
	8	735
*	7	729
*	6	492
*	5	243
*	4	480
*	3	486
*	2	237
	1	0
	0	1

From last: 234 stamps, 0.017secs local = value

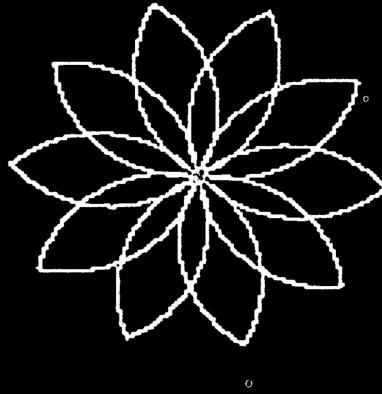
# Logo 1967

```
TO PETAL
```

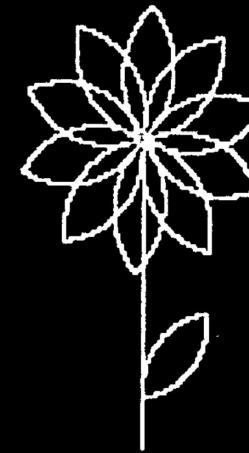
```
QCIRCLE 50  
RIGHT 90  
QCIRCLE 50  
RIGHT 90  
END
```



```
TO NEWFLOWER  
REPEAT 10  
  PETAL  
  RIGHT 360/10  
END
```



```
TO PLANT  
NEWFLOWER  
BACK 50  
PETAL  
BACK 50  
END
```



# Leogo 1997

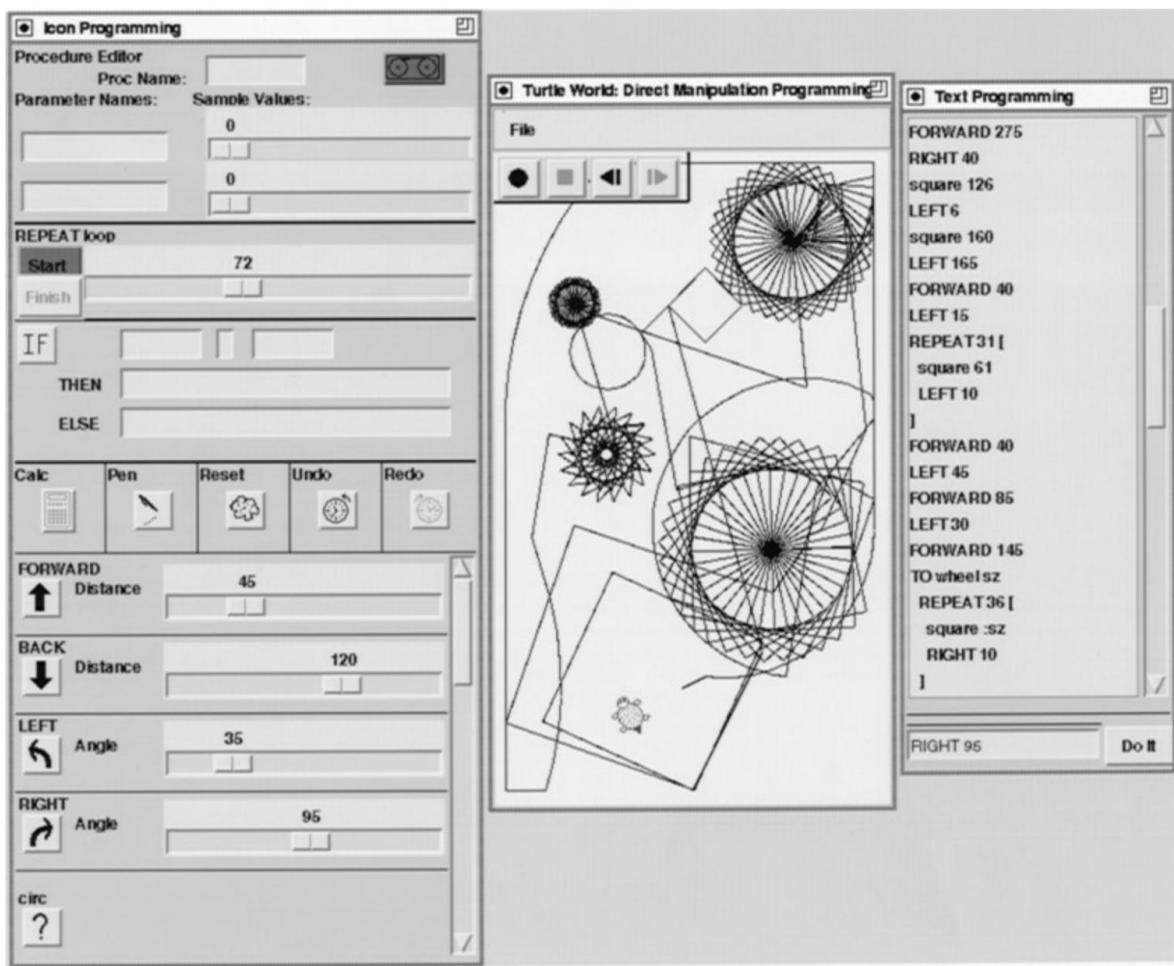


Figure 1. Leogo's three programming paradigms: iconic, direct manipulation, and textual.

## Flogo II 2003

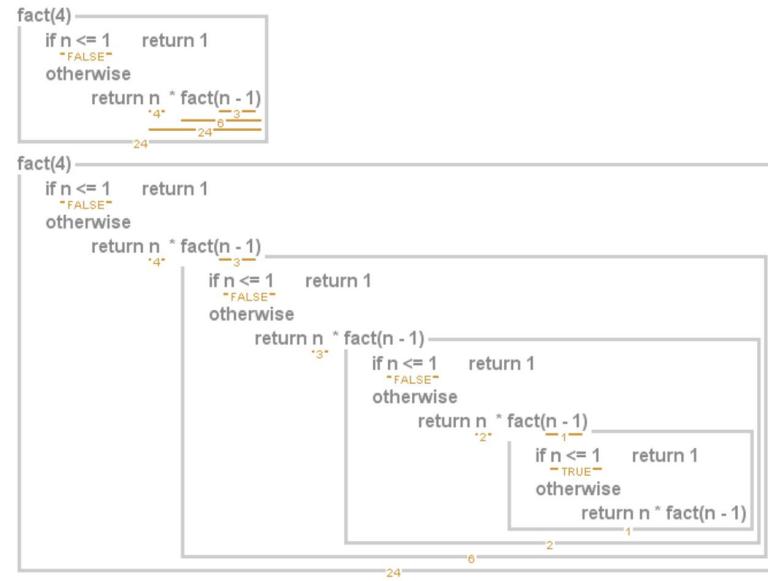


Fig. 5: A recursive function call opened up one level deep (above), and to its full depth (below).

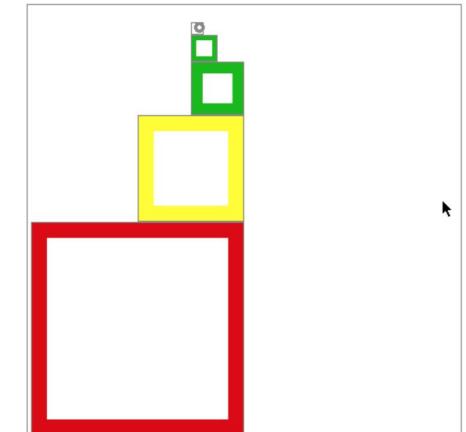
```

fact(4)
if n <= 1    return 1
  "FALSE"
otherwise
  return n * fact(n - 1)
    "4"
    "3"
    "2"
    "1"
      if n <= 1    return 1
        "FALSE"
      otherwise
        return n * fact(n - 1)
          "2"
            if n <= 1    return 1
              "TRUE"
            otherwise
              return n * fact(n - 1)
                "1"

24
6
2
1
  
```

LEFT: {-200} - 200  
 BOTTOM: {-200} - 200  
 SIZE: {400} 400  
 TOP: {200} BOTTOM + SIZE  
 RIGHT: {200} LEFT + SIZE  
 MIDDLE: {0} LEFT + SIZE div 2  
 RECT(TOP + 4,LEFT - 4,BOTTOM - 4,RIGHT + 4,'gray')  
 CX: {-42} FAX(0)  
 CY: {182} FAY(0)  
 CIRCLE(CX,CY,5,'gray',50)  
 WHEN CX < MIDDLE
 □ BOXTOWER(BOTTOM,LEFT,SIZE div 2)
 □ TOP: {0} BOTTOM + SIZE
 □ RIGHT: {0} LEFT + SIZE
 □ FILL: {RED } on resume choose('red','blue','green','yellow')
 □ RECT(TOP,LEFT,BOTTOM,RIGHT,'gray',PCT,FILL)
 □ WHEN CY > TOP and SIZE > 8
 □ MIDDLE: {-100} (LEFT + RIGHT) div 2
 □ WHEN CX < MIDDLE
 □ BOXTOWER(TOP,LEFT,MIDDLE - LEFT)
 OTHERWISE
 □ BOXTOWER(TOP,MIDDLE,RIGHT - MIDDLE)
 □ TOP: {100} BOTTOM + SIZE
 □ RIGHT: {0} LEFT + SIZE
 □ FILL: {YELLOW} on resume choose('red','blue','green','yellow')
 □ RECT(TOP,LEFT,BOTTOM,RIGHT,'gray',PCT,FILL)
 □ WHEN CY > TOP and SIZE > 8
 □ MIDDLE: {-50} (LEFT + RIGHT) div 2
 □ WHEN CX < MIDDLE
 □ BOXTOWER(TOP,LEFT,MIDDLE - LEFT)
 OTHERWISE
 □ BOXTOWER(TOP,MIDDLE,RIGHT - MIDDLE)
 START\_TIME: {317683} on resume tenths\_now()
 AGE: {4 } tenths\_now() - START\_TIME
 PCT: {28 } min(100,AGE % (SIZE div 7))
 OTHERWISE
 □ BOXTOWER(BOTTOM,MIDDLE,SIZE div 2)

Fig. 9. Snapshot of live program execution of BOXTOWER program. Note live nested display of recursive calls.



# Python Tutor 2015

Python 2.7

```
1 def listSum(numbers):  
2     if not numbers:  
3         return 0  
4     else:  
5         f, rest = numbers  
6         return f + listSum(rest)  
7  
8 myList = (1, (2, (3, None)))  
9 total = listSum(myList)
```

[Edit code](#)

→ line that has just executed

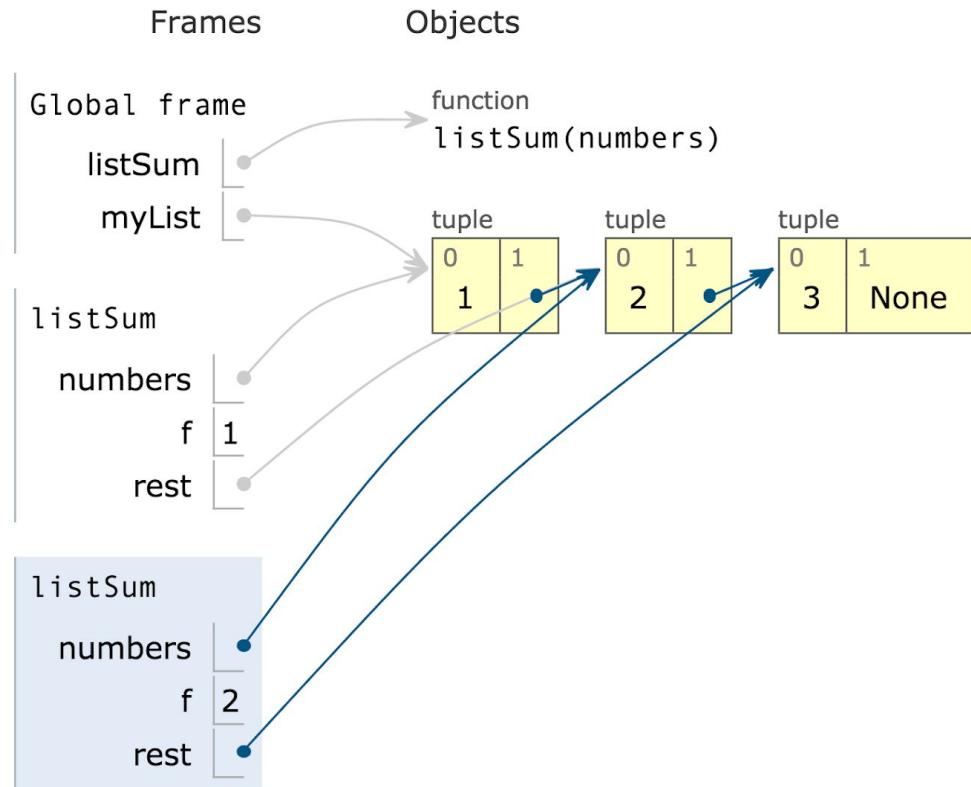
→ next line to execute



< Back

Step 11 of 22

Forward >



# Legible Mathematics 2014

$$(1 + 4 \times 3 + \text{length})^2 = 256$$

$$(1 + 12 + \text{length})^2$$

$$(13 + \text{length})^2$$

$$(13 + 3)^2$$

$$(16)^2$$

$$16^2$$

$$256$$

$$(1 + 4 \times 3 + \text{length})^2 = 256$$



$$(1 + 4 \times 3 + \text{length})^2 = 400$$

16  
↔

$$\text{length} = 3$$

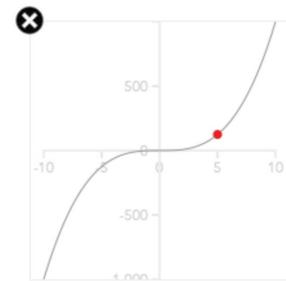
$$(\text{length} + 2)^3$$

x

$$= 3$$

$$= 125$$

y



# LightTable

## 2012

Light Table

```

cube.js* cube.html

requestAnimationFrame( animate );
render();
stats.update();

}

var counter = 0;
var direction = 1;

function render() {
    var time = Date.now() * 0.001; 1377199075137
    mesh.rotation.x = time * 0.25; { _x: 344299768.78025,
                                    _y: 688599537.5605,
                                    _z: 0,
                                    _order: 'XYZ',
                                    _quaternion: [Object] }
    mesh.rotation.y = time * 0.5; 688599537.5685
    renderer.render( scene, camera );
}

```

LightTable

```

Welcome 2013-02-27-light-table-030-experience.md throwaway2.clj*

```

```

;; Anything you type in here will be executed
;; immediately with the results shown on the
;; right.
(ns user) [nil]

(defn neighbours [[x y]] [0 1]
  (for [dx [-1 0 1]
        dy (if (zero? dx) 1
              [-1 1]
              [-1 0 1])]
    [(+ dx x) (+ dy y)])) 0 1 1

(defn step [cells] #[[2 1] ...]
  (set (for [[loc n] (frequencies (mapcat neighbours cells))] #[[2 1] ...
    :when (or (= n 3)
              (and (= n 2) (cells loc))))] #[[2 1] ...
      loc)))

(def board #[[1 0] [1 1] [1 2]]) #[[1 0] ...]

(take 5 (iterate step board)) #[[[1 0] [1 1] [1 2]] [[2 1] [1 1] [0 1]] [[1 0] [1 1] [1 2]] [[2 1] [1 1] [0 1]] [[1 0] [1 1] [1 2]]]

(assoc []) clojure.lang.ArityException: Wrong number of args (1) passed to: core$assoc
  At line:437 clojure.lang.AFn.throwArity
  RestFn.java:412 clojure.lang.RestFn.invoke
NO_SOURCE_FILE:23 user/eval5775

```

```

(defn ->triggers [behs]
  (let [result (atom (transient {}))]
    (doseq [beh behs]
      (t (:triggers (->behavior beh)))
      (swap! result assoc! t (conj (or (get @result t) []) beh)))
      (persistent! @result)))

(defn specificity-sort [xs dir]
  (let [arr #js []]
    (doseq [x xs]
      (doseq
        [clojure.core
         ([seq-exprs & body])
         Repeatedly executes body (presumably for side-effects) with
         bindings and filtering as provided by "for". Does not retain
         the head of the sequence. Returns nil.

         (.push arr #js [(.length (.split (str x) "."))
                        (str x) x])
         (.sort arr)
         (when-not dir (.reverse arr))
         (aloop [i arr] (aset arr i (aget arr i 2)))
         arr))])

```

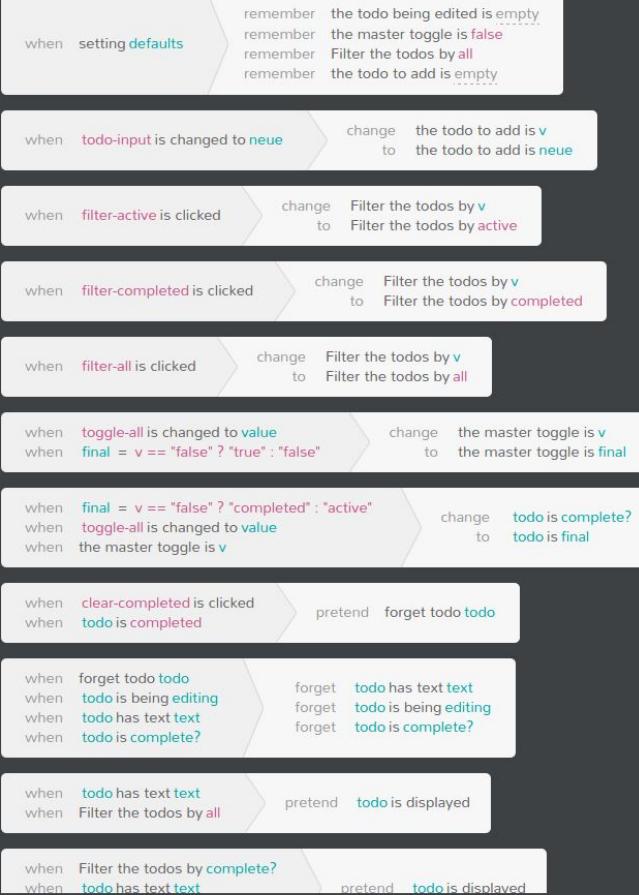
# Eve 2014

[add 100 todos](#) [add 200 todos](#) [mark all completed](#) [remove all](#)

## Todos

What needs to be done?

all active completed



## Flappy Eve

### Setup

Draw the game world!

### Game menus

Score calculation

Start a new game

### Drawing

Player

Obstacles

### Game Logic

Obstacles

Flapping the player

Scroll the world

Collision

## Flappy Eve

When a player starts the game, we commit a `#world`, a `#player`, and some `#obstacles`.

These will keep all of the essential state of the game. All of this information could have been stored on the world, but for clarity we break the important bits of state into objects that they effect.

- The `#world` tracks the distance the player has travelled, the current game screen, and the high score.
- The `#player` stores his current y position and (vertical) velocity.
- The `#obstacles` have their (horizontal) offset and gap widths. We put distance on the world and only keep two obstacles; rather than moving the player through the world, we keep the player stationary and move the world past the player. When an obstacle goes off screen, we will wrap it around, update the placement of its gap, and continue on.

### Setup

Add a flappy eve and a world for it to flap in:

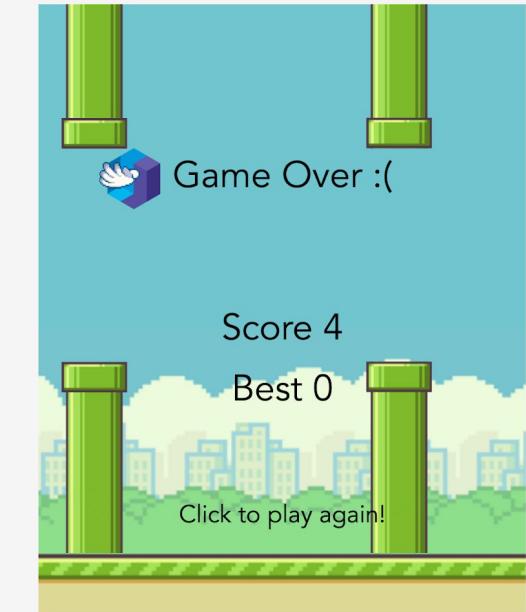
```
commit
  [#player #self name: "eve" x: 25 y: 50 velocity: 0]
  [#world screen: "menu" frame: 0 distance: 0 best: 0 gravity: -0.061]
  [#obstacle gap: 35 offset: 0]
  [#obstacle gap: 35 offset: -1]
```

Next we draw the backdrop of the world. The player and obstacle will be drawn later based on their current state. Throughout the app we use resources from @bhauman's flappy bird demo in clojure. Since none of these things change over time, we commit them once when the player starts the game.

### Draw the game world!

```
search
  world = [#world]

commit @browser
  world <- [#div style: [user-select: "none" -webkit-user-select: "none" -
  moz-user-select: "none"] children:
    [#svg #game-window viewBox: "10 0 80 100", width: 480 children:
      [#rect x: 0 y: 0 width: 100 height: 53 fill: "rgb(112, 197, 206)" sort:
```



# ConQuer

1997

Q2 ✓Employee

  +- drives Car

    +- works for ✓Branch

Q3 ✓USbranch

  +- **not** achieved Rank = 1 in Year < 1998

    +- **is** Branch

      +- is headed by Employee

  +- has ✓EmployeeName

    +- **maybe** drives ✓Car

Q4 ✓Employee<sub>1</sub>

  +- lives in City<sub>1</sub>

  +- was born in Country<sub>1</sub>

  +- supervises Employee<sub>2</sub>

    +- lives in City<sub>1</sub>

    +- was born in Country<sub>2</sub> <> Country<sub>1</sub>

Q5 ✓Employee

  +- owns Car<sub>1</sub>

  +- **not** drives Car<sub>1</sub>

    +- **count**(Car<sub>1</sub>) **for** Employee > 1

# SIEUFERD 2016

Ethanol-Related Organizations <						
OpenSecrets Name	f <sub>x</sub> Total Amounts in 2012-dollars	Lobbying Reports <`		CPI Lookup		
		Type	Amount	Year	Consumer Price Idx	
Low Carbon Synthetic Fuels Association	41,170	q2	40,000	2011	0.9716	
Maple Etanol SRL	74,870	q1 q4 q3 q2	10,000 20,000 30,000 10,000	2010 2009 2009 2009	0.9560 0.9315 0.9315 0.9315	
Algal Biomass Org	=sum([Amount]) / [Consumer Price Idx])				1.0000 1.0000 0.9716 0.9716 0.9560	
American Council on Renewable Energy	246,962	q1t q1 q4 q4 n <sub>2</sub>	20,000 30,000 30,000 40,000 nn nn	2009 2009 2008 2008 2008	0.9315 0.9315 0.9313 0.9313 0.9313	

Figure 1: Spreadsheet-style formula editing in a nested relational result. The query structure is encoded in the table header, which shows three joined table instances (bold labels), one-to-many relationships ( $\leftarrow$ ), sorting ( $\triangleright$ ), active filters ( $\text{Y}$ ,  $\text{K}$ ), and a formula ( $f_x$ ).

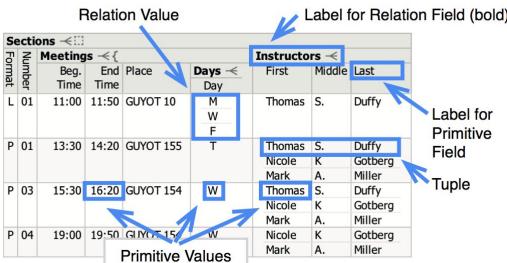
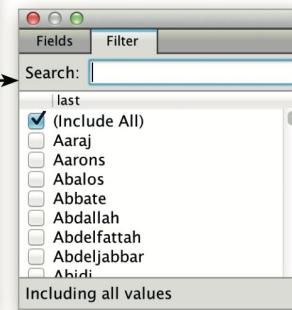
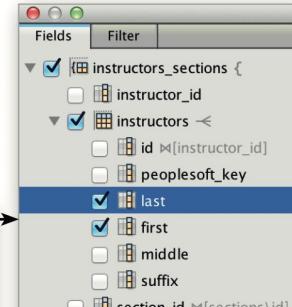


Figure 3: Terminology of the nested relational data model, illustrated on a nested table layout.

Niednagel														
courses <			readings <			sections <			meetings <			instructors_sections <		
title	readings	sections	format	number	day	begin_time	end_time	time	instructors	instructors	sections			
Roman Art	Roman Art	Roman Art	L 01	T	14:30	15:20	Th	14:30	Meyer	Hugo				
	Art and Illusion		P 01	W	19:30	20:20			Meyer	Hugo				
Comedy	The Miser	The Miser	L 01	M	11:00	11:50			Lachman	Kathryn				
	A Flea in Her Ear		P 02	W	12:30	13:20			Niedn					
	Art		P 03	W	13:30	14:20			Barka					
			P 04	F	11:00	11:50			Fiske					
			P 05	W	14:30	15:20			Fiske					
			P 06	Th	11:00	11:50			Hasty					
Russian Drama	Little Tragedies	Little Tragedies	S 01	T	11:00	12:20								
	The Seagull			Th	11:00	12:20								
	The Duck Hunt													
American Politics			L 01	M	11:00	11:50			Troup					
			P 01	W	11:00	11:50			Gada					
			P 02	M	13:30	14:20			Gada					
			P 01	W	13:30	14:20			Gada					
			P 03	M	14:30	15:20			Gada					
			P 04	W	14:30	15:20			Troup					
			P 05	Th	09:00	09:50								

**Result area:** Displays the currently open query and its nested relational result. Labels and formulas can be edited using a spreadsheet-like cursor.

**Context menu:** Exposes a complete set of query manipulation actions, and serves as a legend for all icons that can appear in the result header.



**Formula bar:** Shows the  $\text{label}$ ,  $\text{value}$ , or  $\text{formula}$  under the selected cell.

**Result header:** Visually encodes both the structure of the query and the schema of its result. Icons indicate query-related state associated with each field in the schema.

**Field selector:** Pop-up displaying a tree representation of the query structure, including exact join conditions, centered around the selected field. The selector includes previously hidden fields as well as fields that can be reached through joins over known foreign key relationships.

**Filter popup:** Allows the user to associate a filter with the currently selected field. The list of values available to filter on is generated automatically using a separate database query. Filters may be associated with either primitive fields or relation fields.

Figure 2: The SIEUFERD query interface. To create queries, users start from a simple tabular view of a table in the database and add filters, formulas, and nested relations. The integrated result and query representation is displayed continuously as the user interacts with the data. The particular query above instantiates six database tables (one per nested relation), contains five joins (each child relation against its parent), and is evaluated using five generated SQL queries (one for each one-to-many relationship  $\leftarrow$ ). This query was constructed purely by checking off the appropriate fields and foreign key relationships in the field selector.

# Object Spreadsheets 2016

Class:Section:Enrollment:scheduledSlot : auto (Person:Slot)

computed by formula [Editability / triggers](#) [Global macros](#)

Formula: {\$Person.Slot by scheduledEnrollment}

[Save](#) [Cancel](#)

Undo  
Redo  
Cleanup

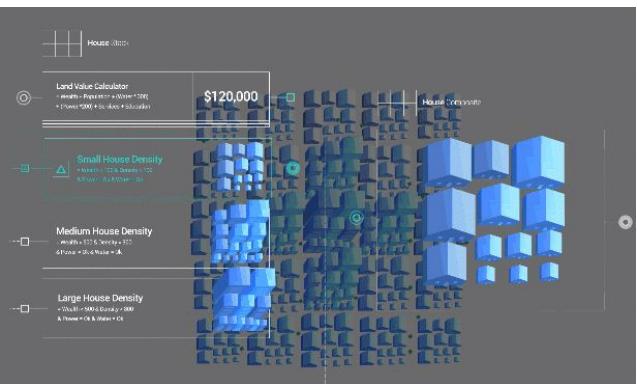
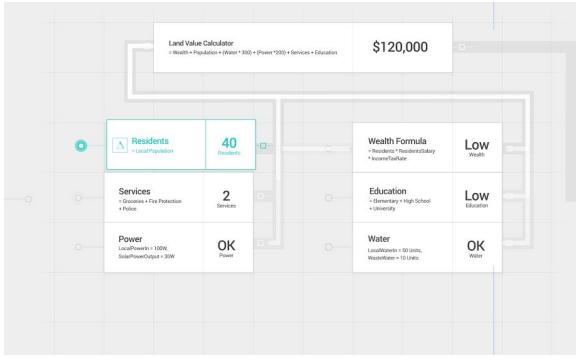
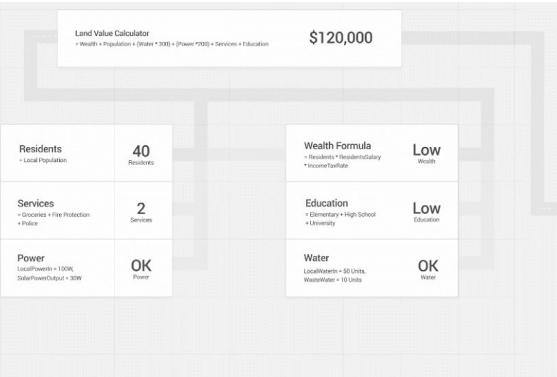
Person			Slot			Enrollment					
	name	roles	parents	time	scheduledEnrollment	valid	teacher	student	scheduledSlot	valid	valid
*	Snape	teacher	Person	* text	Enrollment	bool	* Person	* Person	Slot	bool	bool
*				2014-12-16 13:00	Seamus in 6.820	true	Snape	Ronald	Snape @ 2014-12-16 13:45	true	true
				2014-12-16 13:30		true		Ginevra			
				2014-12-16 13:45	Ronald in 6.170	true		Ronald			
*	Flitwick	teacher	Person	* text	Ginevra in 6.005	true	6.170	Potions		true	
				2014-12-17 13:00		true	*				
				2014-12-17 14:00		true	*	Snape			
*	Ronald	student	Molly				*	Ronald			
*	Ginevra	student	Molly				*	Ginevra			
*	Seamus	student	Augustus				*	Seamus	Snape @ 2014-12-16 13:00	true	
*	Molly						*	Seamus			
*	Augustus						*	Flitwick	Ginevra Flitwick @ 2014-12-17 13:00	true	
									Flitwick		
										Seamus	

```

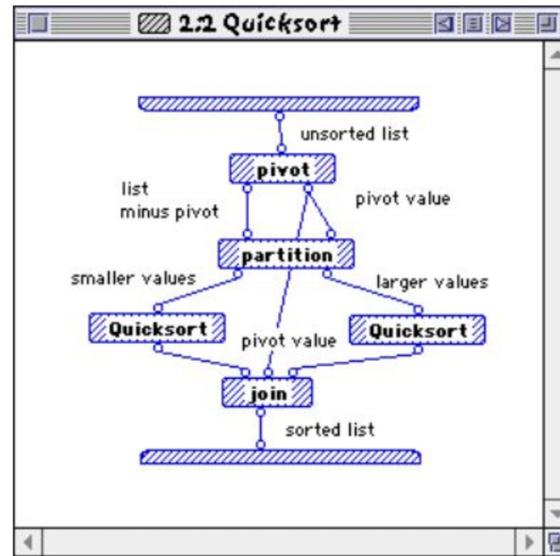
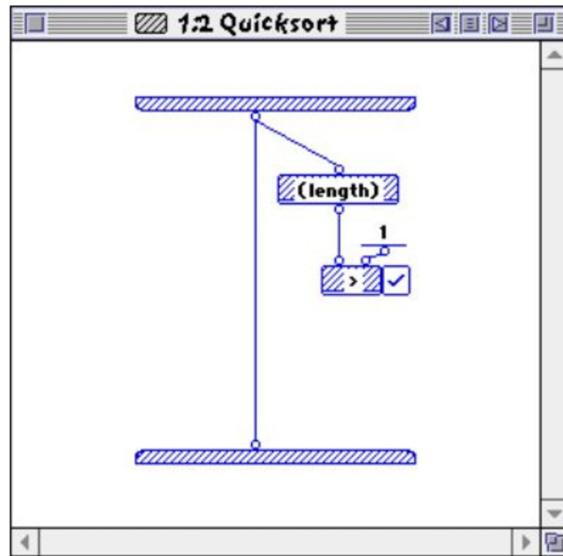
Enrollment.scheduledSlot ≡ Enrollment.{$Person.Slot by scheduledEnrollment}
Slot.valid           ≡ count(time) = 1 && count(scheduledEnrollment) <= 1 &&
                      scheduledEnrollment.Section.teacher = Person
Enrollment.valid    ≡ count(scheduledSlot) <= 1
$valid              ≡ !(false in {$Person.Slot.valid, $Class.Section.Enrollment.valid})

```

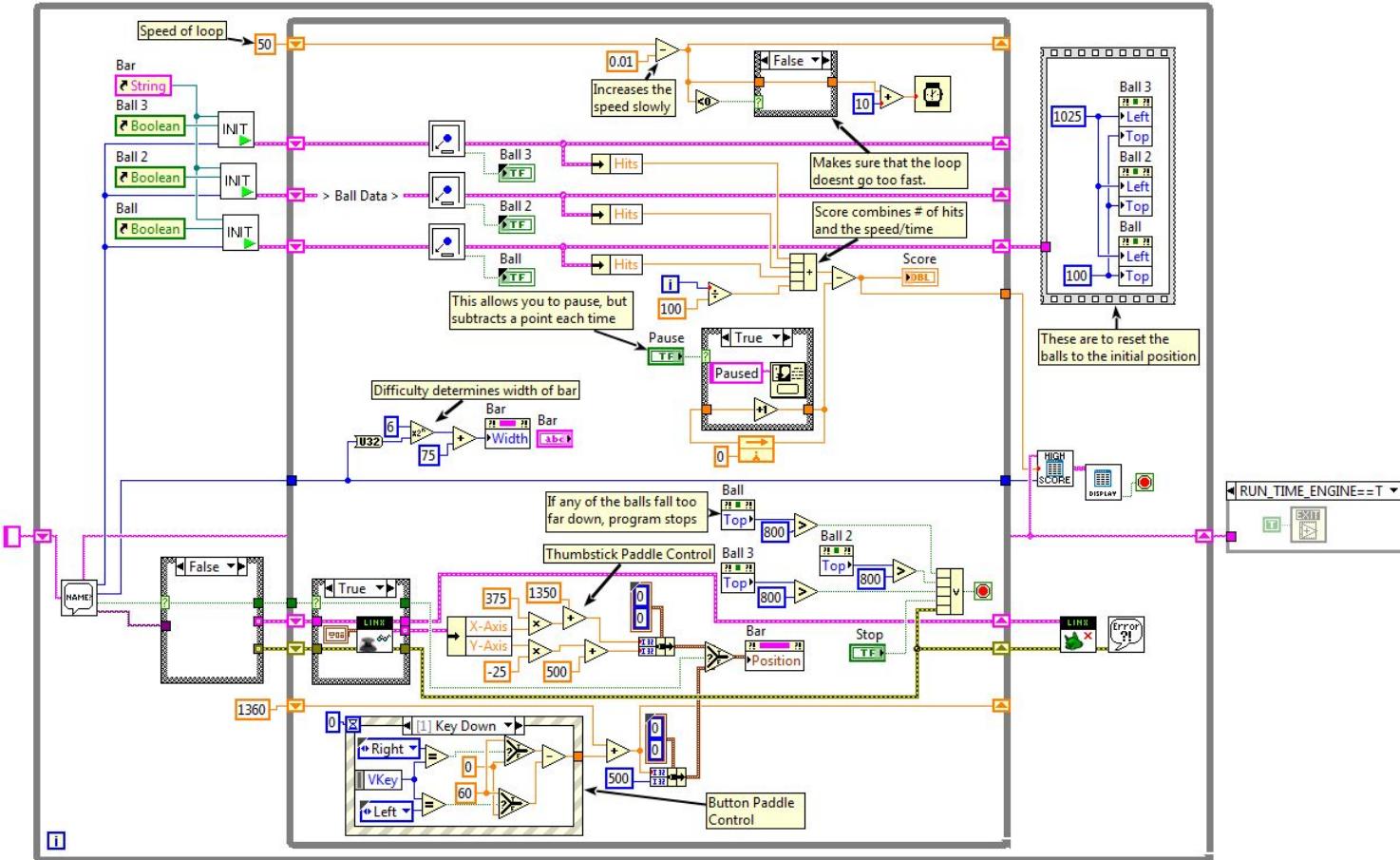
# Grid 2013



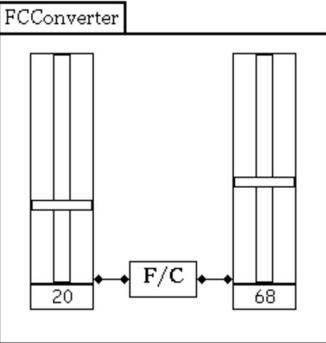
# ProGraph 1983



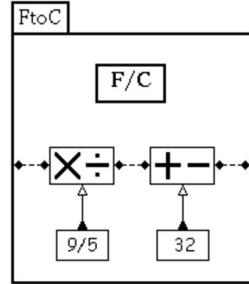
# LabVIEW 1986



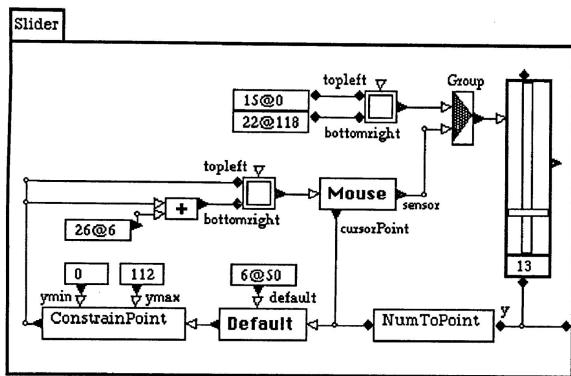
# Fabrik 1988



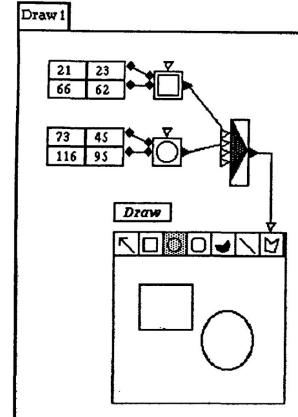
**Figure 2a.** Bidirectional diagram using two Slider controls to achieve a Fahrenheit-to-Centigrade converter.



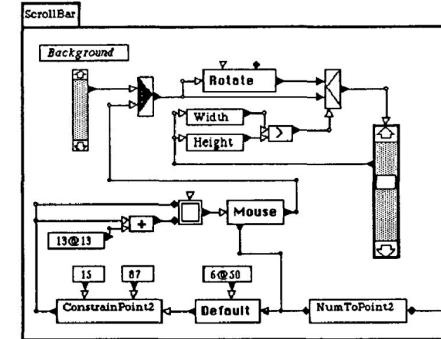
**Figure 2b.** Internal diagram for the F/C component used in the diagram at left.



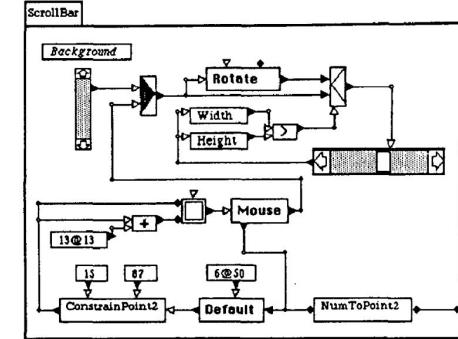
**Figure 3.** A Fabrik diagram computes the image for the slider in figure 2. The Mouse component sensitizes the slider image to support input as well as output.



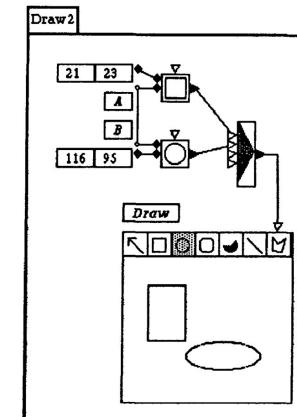
**Figure 5a.** The Draw component automatically lays out diagrams as the user creates a drawing.



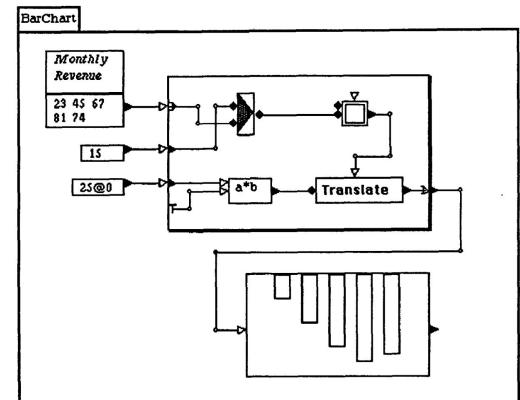
**Figure 4a.** A simple scrollbar diagram. Logic is provided for rotating when the image is wider than high.



**Figure 4b.** Demonstrating the rotation logic. All display and mouse-tracking operations are properly scaled and rotated.

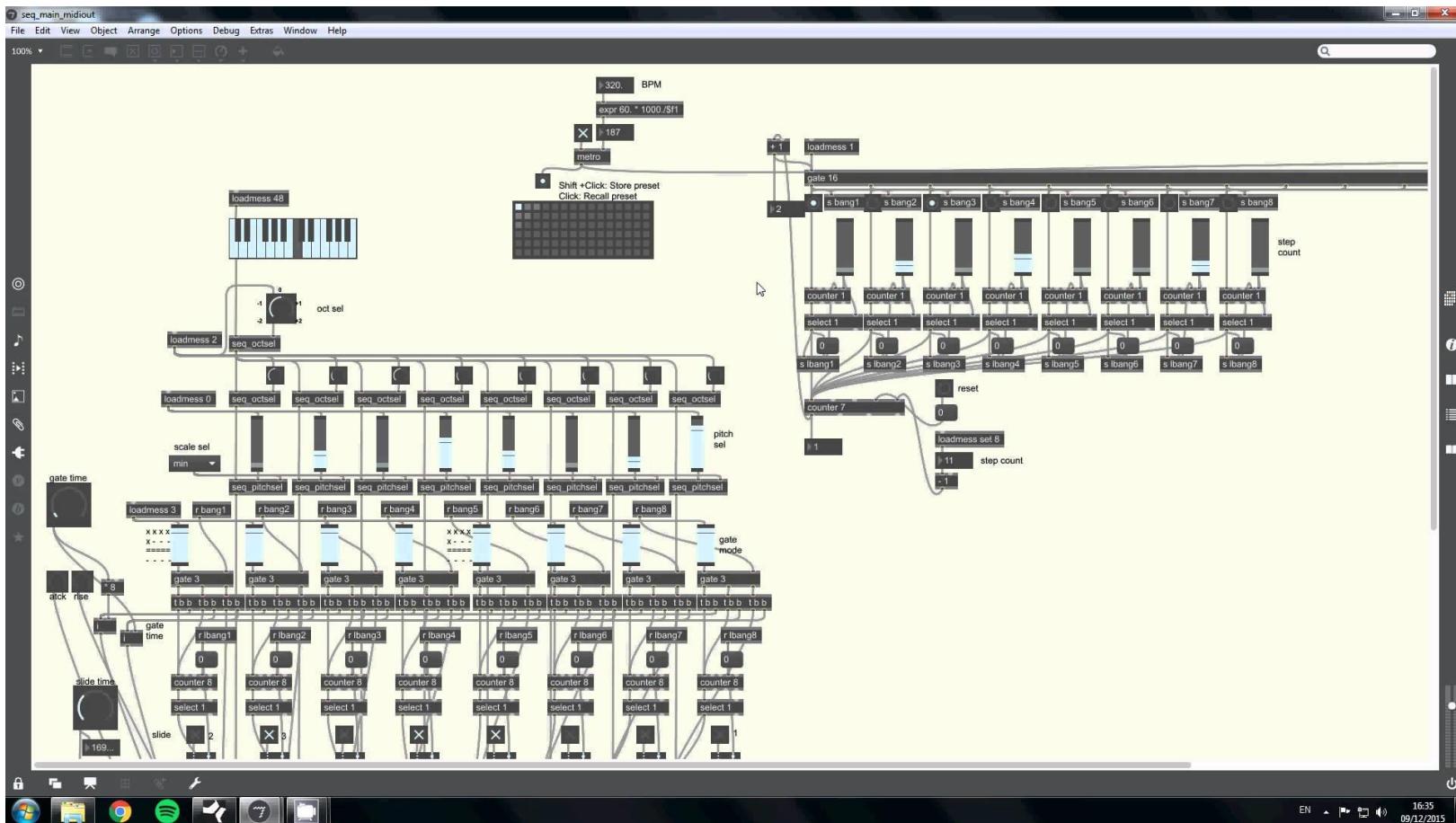


**Figure 5b.** By editing the generative diagram, the top-left of the oval is tied to the bottom-right of the rectangle.

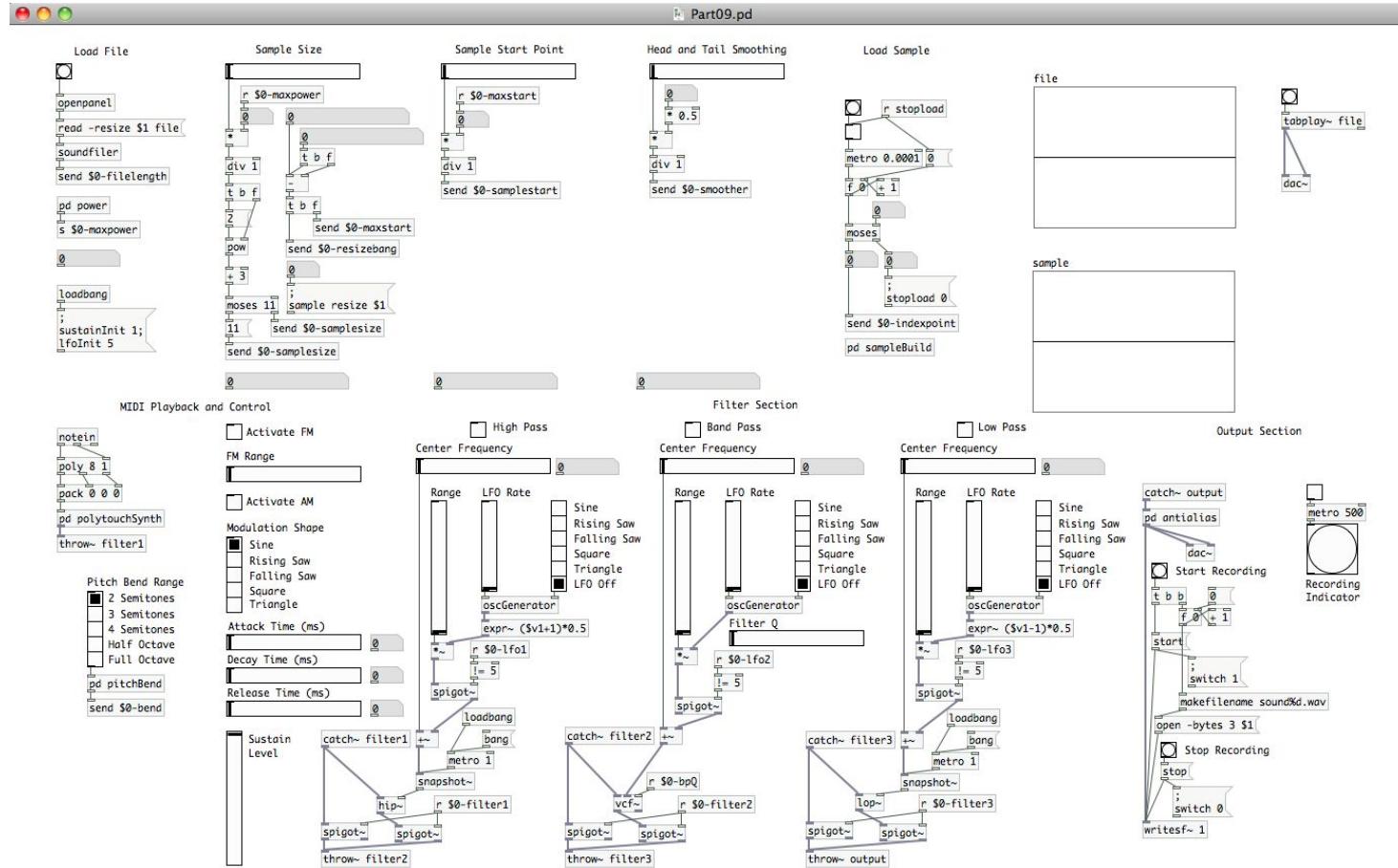


**Figure 6.** Streaming gateways provide iterative capability. Here numbers are converted to rectangles, resulting in a simple bar chart.

# MaxMSP 1990



# Pure Data 1996



# vvvv 1998

Project Explorer

Spray.cs+ C:\Users\Joreg\Desktop\vvvvs\vvvv\_45beta31.2\_x86\lib\nodes\plugins\Spray\

```

14  namespace VVVV.Nodes
15  {
16      #region PluginInfo
17      [PluginInfo(Name = "Spray",
18                  Category = "Animation")]
19      #endregion PluginInfo
20      public class Explosion : IPluginEvaluate
21      {
22          fields & pins
23
24          //called each frame by vvvv
25          public void Evaluate(int SpreadMax)
26          {
27              //update particles and remove timed out particles
28              for (int i = FParticles.Count - 1; i >= 0; i--)
29                  if (!FParticles[i].Update(FHDEHost.GetCurrentTime()))
30                      FParticles.RemoveAt(i);
31
32              for (int i = 0; i < SpreadMax; i++)
33              {
34                  //add new particles
35                  if (FBang[i])
36                  {
37                      for (int j = 0; j < 4; j++)
38                      {
39                          var vel = new Vector3D(FRandom.NextDouble() - 0.1f,
40                          vel.z = 0;
41                          var acc = new Vector3D(0, FAcc[i], 0);
42                          FParticles.Add(new Particle(FHDEHost.GetCurrentTime(),
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
    
```

Spray.v4p \* C:\Users\Joreg\Desktop\vvvvs\vvvv\_45beta31.2\_x86\girlpower\

vvvv ->

vvelcome,  
left click the r  
note how the :  
realized as a c  
don't show th  
delete args.tx  
F1 for some b

DirectX Renderer

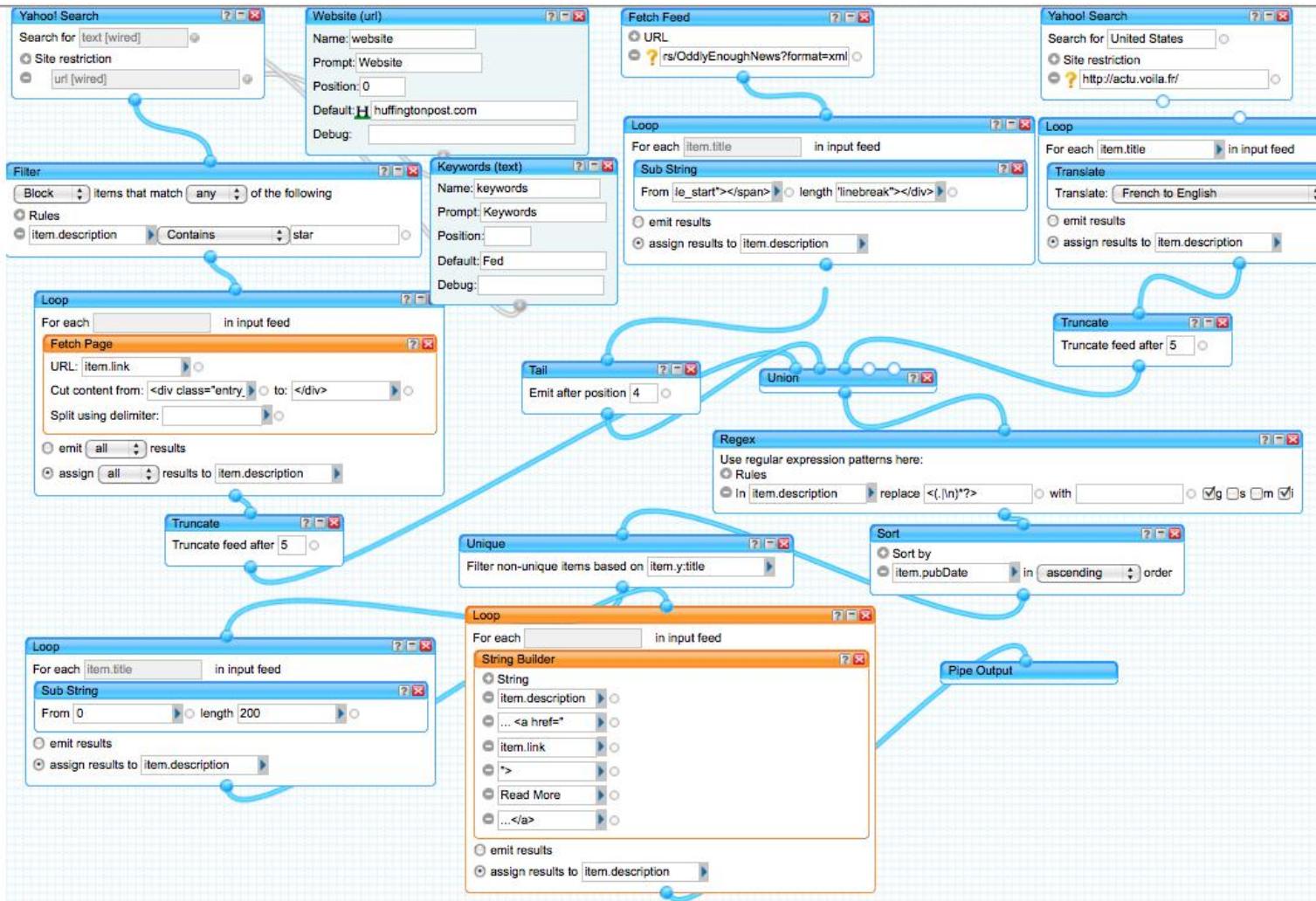
Spray (Animation)

Descriptive Name
InputXYZ
Bang
Acceleration
Max Lifetime
OutputXYZ
Age
ID

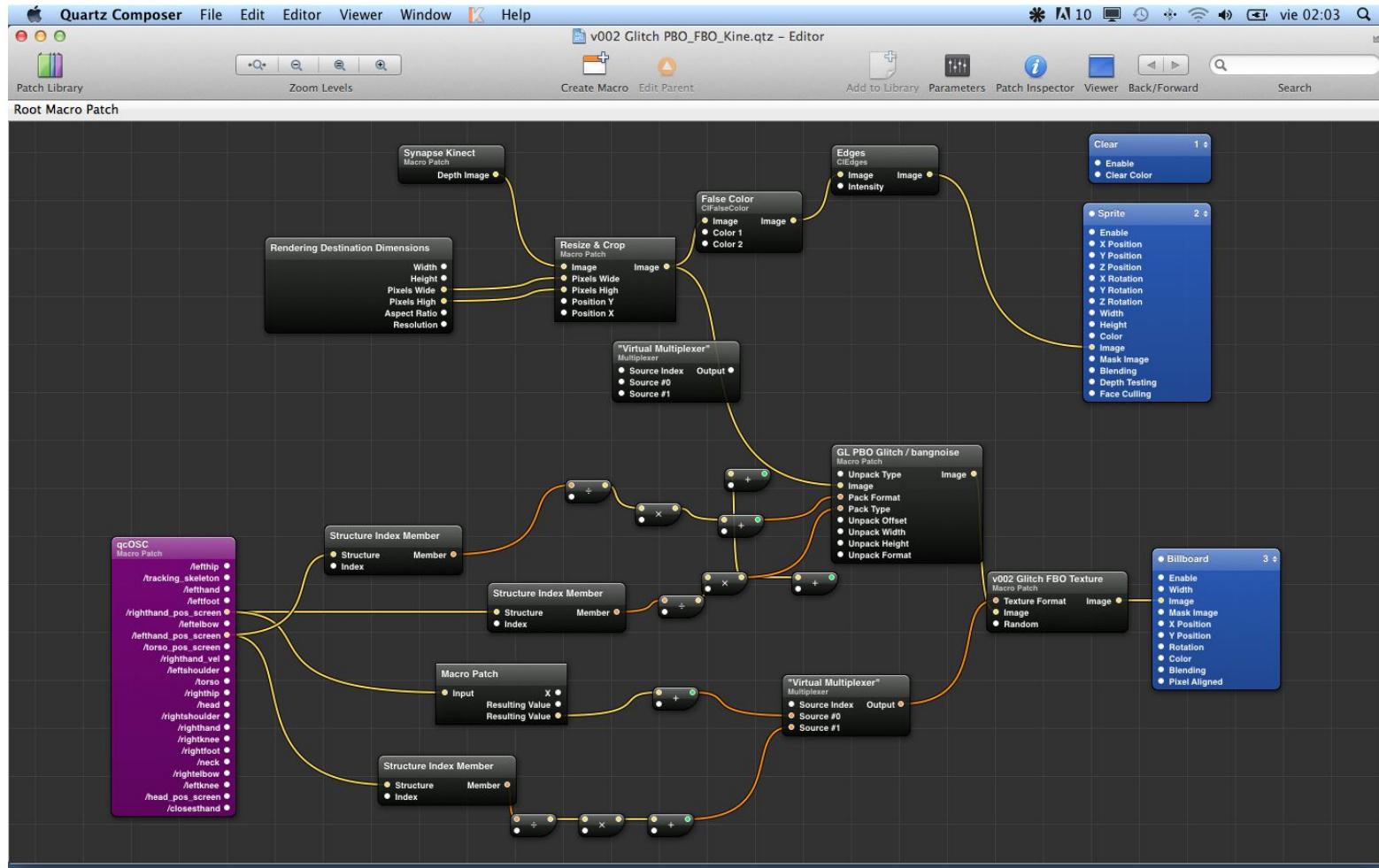
create your own dynamic  
doubleclick in a patch, tv  
available and clone them

# Yahoo Pipes

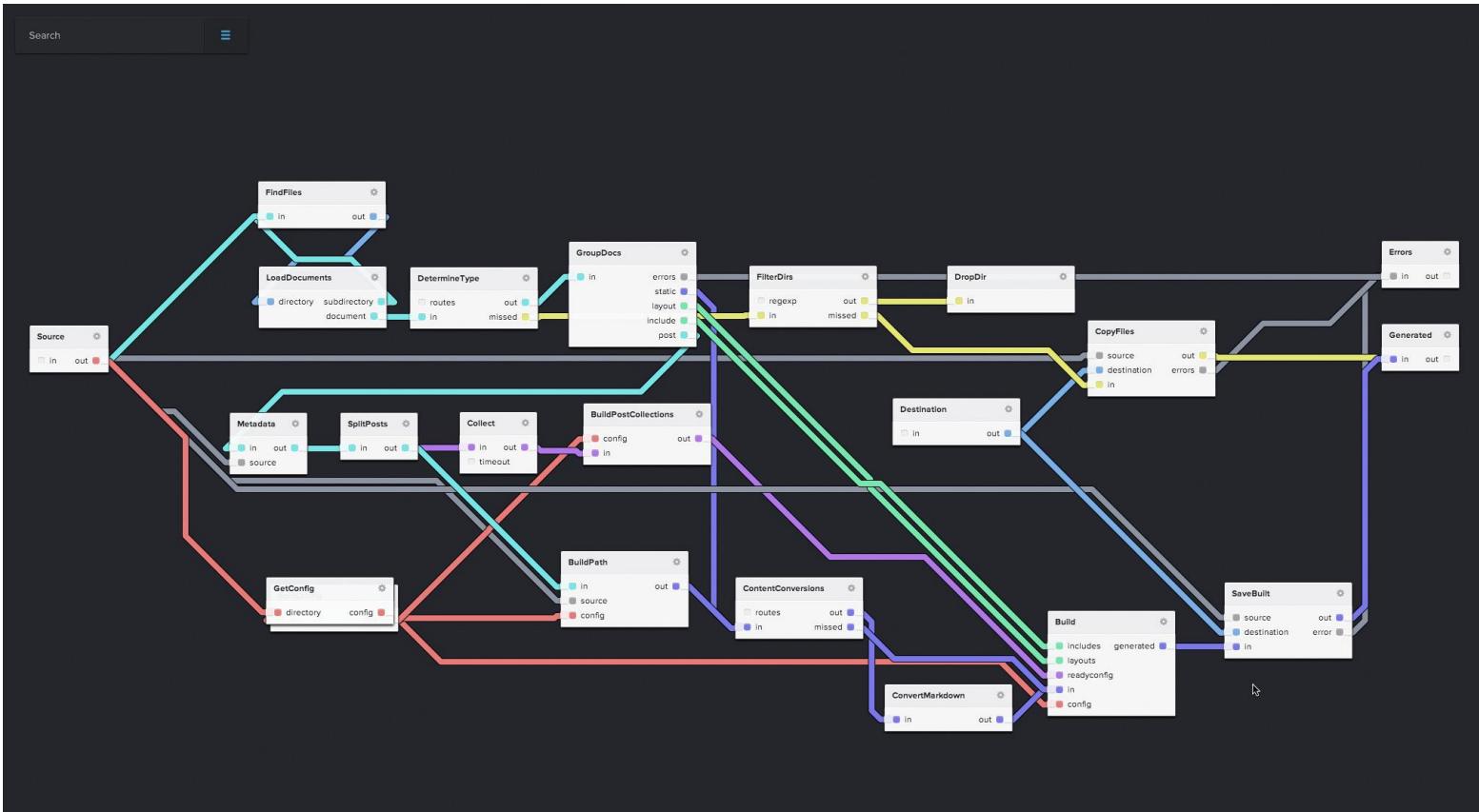
## 2007



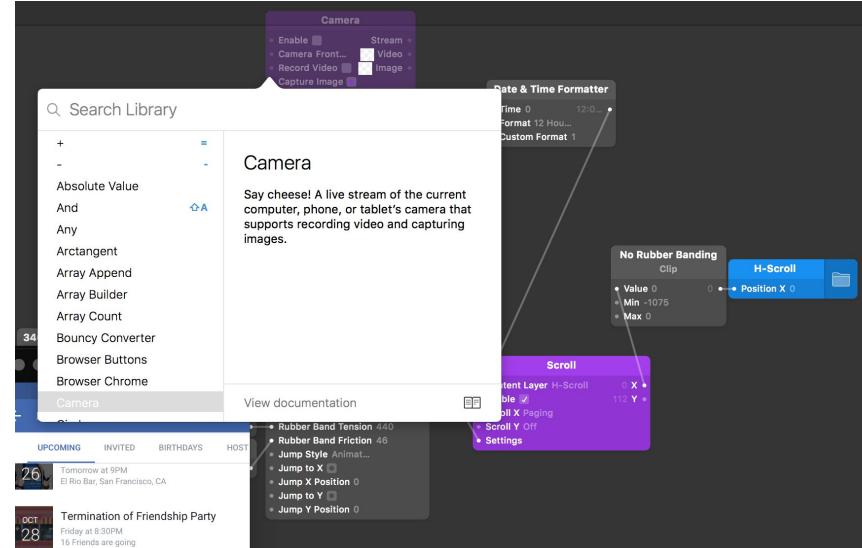
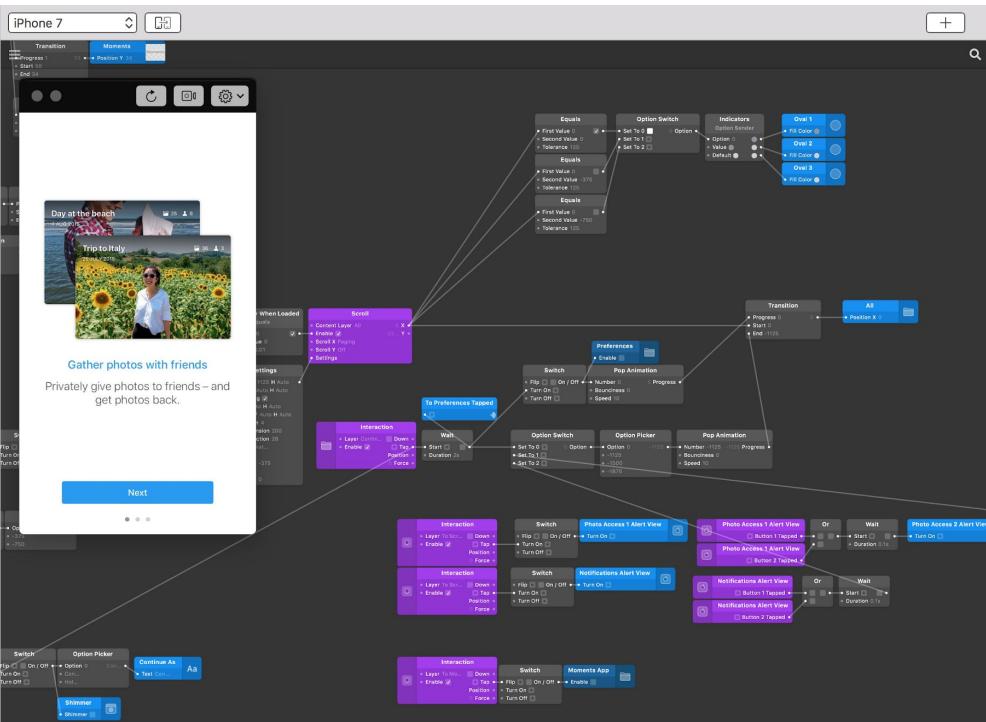
# Quartz Composer 2005



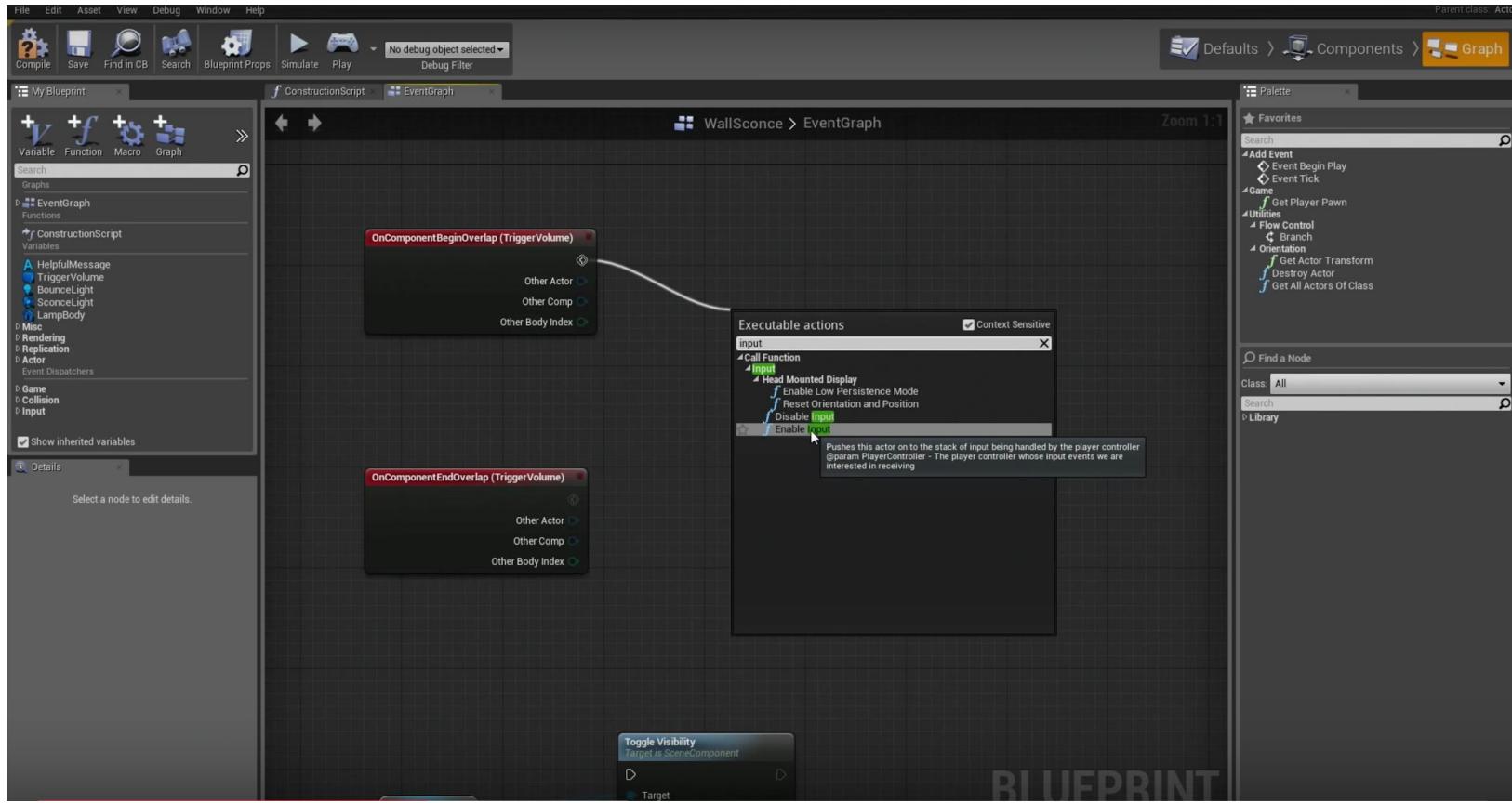
# NoFlo 2014



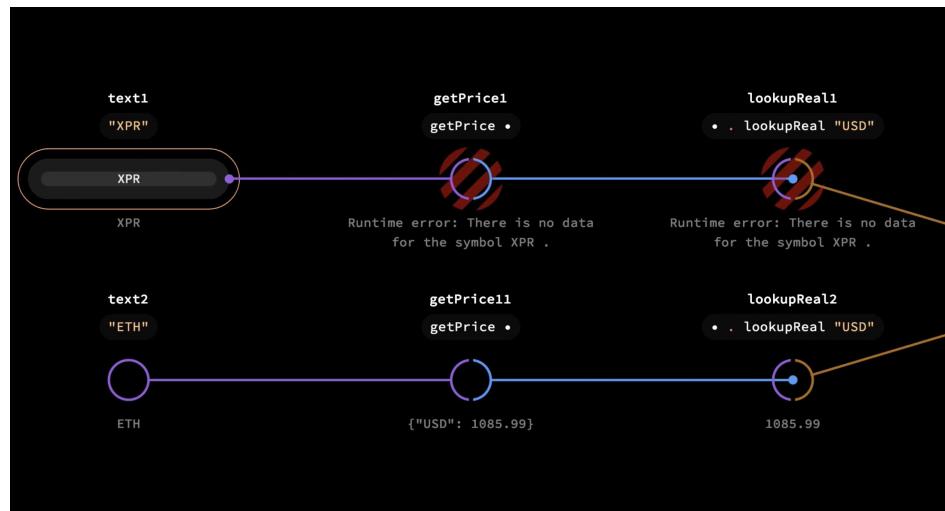
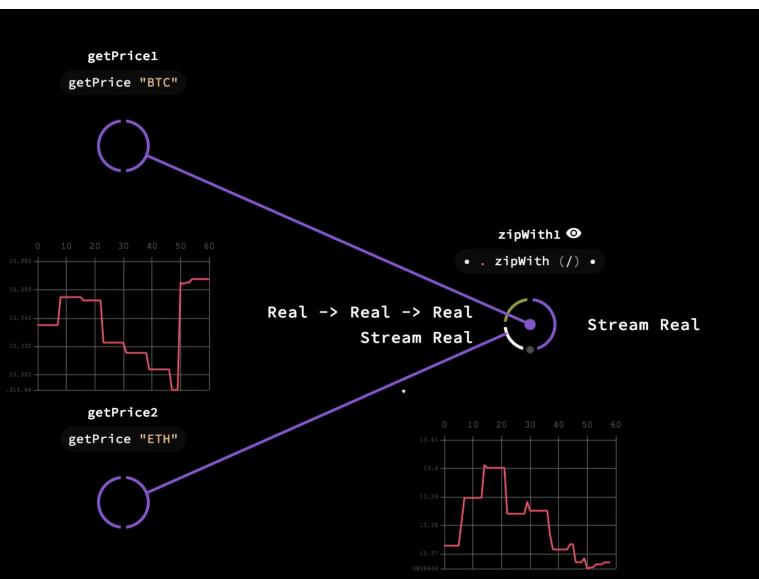
# Origami 2017



# Unreal Blueprint 2014 (contextual autocomplete)



# Luna 2018



# Early structure editors

DEdit of function FACT

```
(LAMBDA (X) (* mjs " 7-Oct-85 16:04")
  (if (LESSP X 2)
    then 1
    else (TIMES X
      (FACT_(SUB1_X)))))
```

Edit buffer

```
(FACT (DIFFERENCE X 1))
```

EditOps

- After
- Before
- Delete
- Replace
- Switch
- ()
- () out
- Undo
- Find
- Swap
- Reprint
- Edit
- EditCom
- Break
- Eval
- Exit

Interlisp DEdit 1973

main

```
program <identifier>;
var
  <identifier> : <type>;
begin
  i { NOT DECLARED } := 1;
  while <exp> do
    <statement>
end.
```

Positioned at exp

Cornell Program Synthesizer/  
Synthesizer Generator 1981

# Pecan 1984

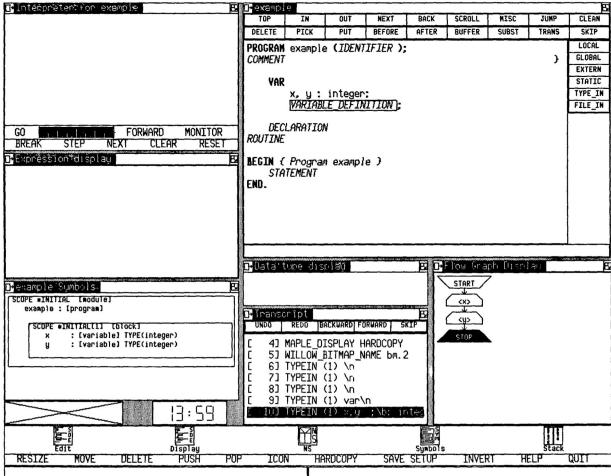


Figure 2: PECAN Display Showing Editing

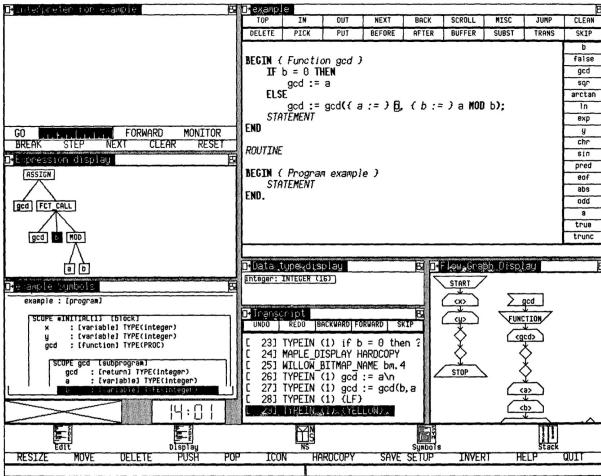


Figure 3: PECAN Display Showing Semantic Views

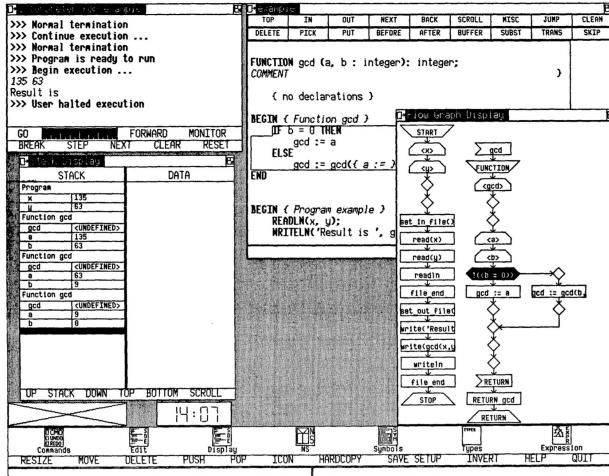
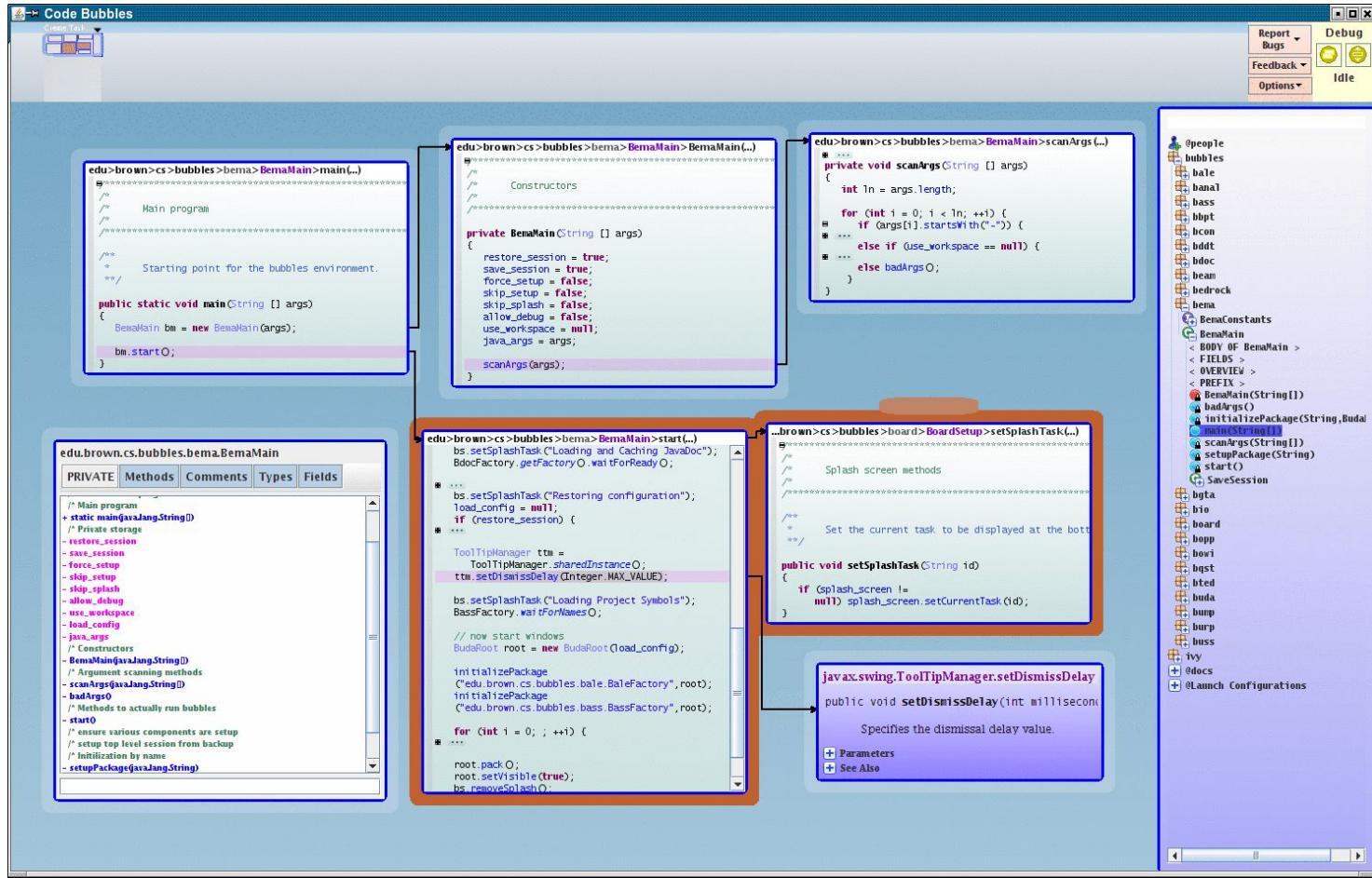


Figure 4: PECAN Display Showing Execution

# Code Bubbles 2010



# Patchworks 2014

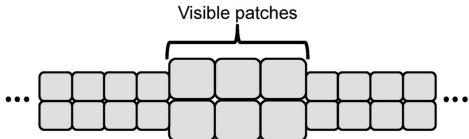


Figure 5. The conceptual ribbon of patches.

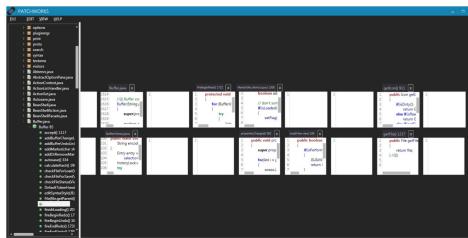


Figure 6. Patchworks' ribbon view, which provides a bird's eye view of the ribbon.

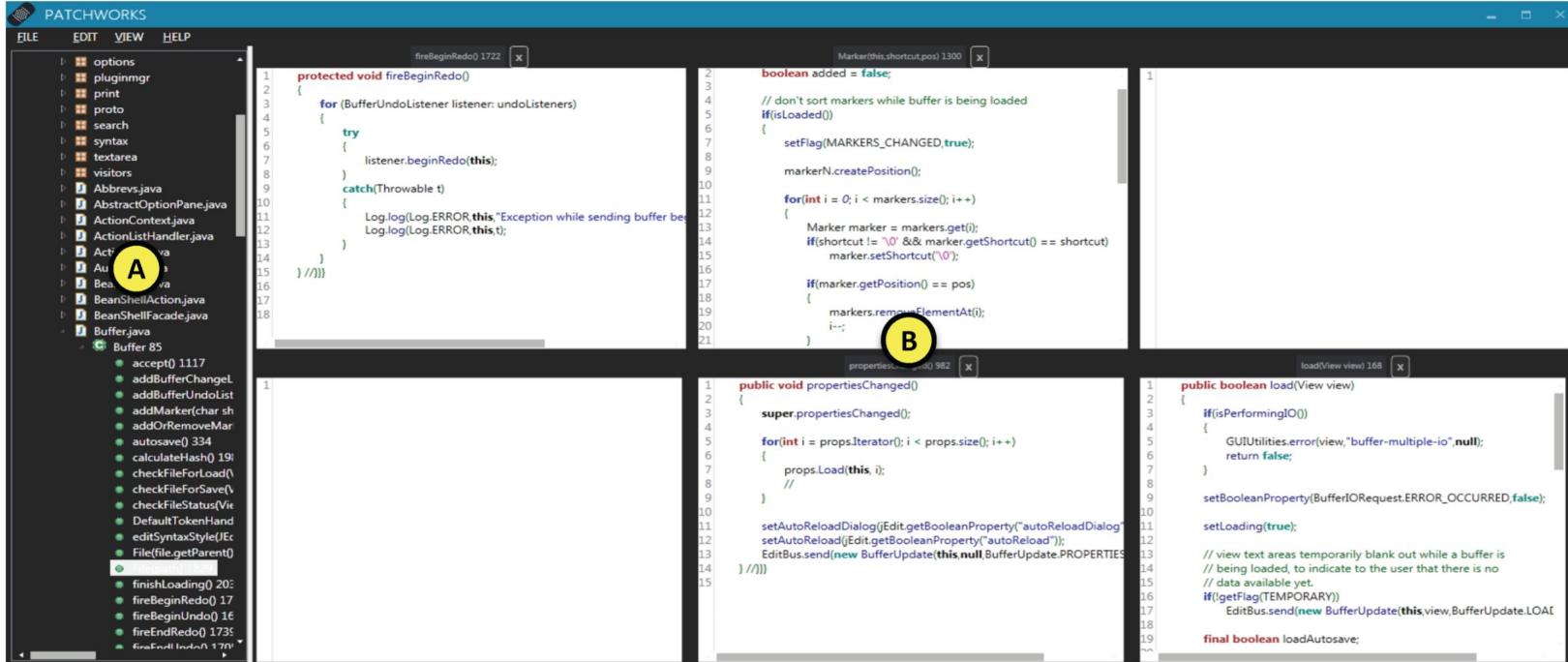


Figure 3. The Patchworks editor, including (A) a package explorer and (B) a 3×2 patch grid. Four of the patches contain code fragments and two are empty.

# DrRacket 1997

enum-slides.rkt (define ...) enum-slides.rkt

Check Syntax ⚡ Debug ⚡ Macro Stepper ⚡ Run ⚡ Stop

2: enum-time.rkt

```
1 #lang at-exp racket
  center-color)
365
366
367 (blank)
368
369 (force pair-enum-animation)))
370
371 (define (pair-enum)
372   (slide (linewidt ... (grid cons/e 8 63 500 24 #:arrows? #t))
373          @#(This series of arrows shows how the result
374         bijection traces out a path in the space of the pairs
375         of the elements of the two input bijections.
376
377 That is, imagine that the elements of T are
378 going horizontally along the top (starting at
379 the origin). Then imagine that the elements
380 of T' are going vertically down the slide. And
381 in each cell is a pair. The resulting bijection will
382 follow this path. Roughly, it is tracing out a series
383 of ever widening squares in the plane.
384
385 You may have heard of the Cantor pairing function?
386 It is similar to this one, but traces out the edges
387 of ever larger triangles, going diagonally. That one
388 turns out not to generalize to n-dimensional functions
389 while this one does. I could write the
390 reverse part of the bijection, which is why we use
391 this one. There is a lot more to say here, but for
392 today, lets just stick with the basic idea.
393 )))
394
395 (define (grid cons/e count num-points size arrow-head-size #:arrows? arrows?)
396   (define prs (cons/e natural/e natural/e))
397   (define base
398     (colorize
399       (dc (lambda [x dc dx dy]
400             (for ([i (in-range 1 count)])
401               (send dc draw-line
402                 (+ dx (* i / size count))
403                 dy
404                 (+ dx (* i / size count))
405                 (+ dy size)))
406               (send dc draw-line
407                 (+ dy (* i / size count))
408                 (+ dx size)
409                 (+ dy (* i / size count)))))))
410
411   size)
412   size)
413   "gray"))
414   (hb-append
415     (apply
416       vc-append
417       (for/list ([i (in-range count)])
418         (define txt (t (format "~a" i)))
419         (cc-superimpose
420           (blank 0 (/ size count))
421           (refocus (vc-append
422             (txt
423               (if (= i (- count 1))
424                 (t " ")
425                 (blank)))
426             txt))))))
427   (vc-append
428     (apply
429       hc-append
430       (for/list ([i (in-range count)])
431         (define txt (t (format "~a" i)))
432         (cc-superimpose (blank (/ size count) 0)
433                         (refocus
434                           (hb-append
435                             (txt
436                               (if (= i (- count 1))
```

Welcome to DrRacket, version 6.0 (lm).

Language: at-exp racket, with debugging; memory limit: 256 MB.

> (for/list ([i (in-range 10)])  
 (from-nat  
 (pam/e bt-to-pict  
 av/e  
 #:contract pict?))  
 1))

' | )

Follow IRequires  
Follow PlanET requires  
Name length: Long  
Refresh

# Intentional Software 2002

**Documents**

- Add Document Type
- Document Type Fund Report**
  - Close
  - Add Fund Report Document
- Options
- Parameters
- Add Parameter
 

#	Name	Type	Description	Default value	Optional
1	Country	Country			<input type="checkbox"/>
2	Fund	Fund			<input type="checkbox"/>
3	Publishing Date	Date			<input type="checkbox"/>
4	Portfolio Commentary	Text			<input type="checkbox"/>
5	Cumulative Performance Graph	Text			<input type="checkbox"/>
- Procedures
- Template

**FUND's Title\***  
**FUND's Name\***

**FormatDate(Publishing Date)**

**Fund Description**

**Fund's Description\***

**Investment Objective**

**Fund's Investment Objective\***

**Country Information**

AMY<sup>\*</sup> ROUND(Country's Excel Data's AMY \* 100)%  
 APO<sup>\*</sup> ROUND(Country's Excel Data's APO \* 100)%  
 SDHY<sup>\*</sup> ROUND(Country's Excel Data's SDHY \* 100)%  
 Spangler<sup>\*</sup> ROUND(Country's Excel Data's SPANGLER \* 100)%

**Portfolio Commentary**

**Portfolio Commentary\***

**Calendar Year Performance (%)**

2006 2007 2008 2009 2010 YTD

ASSIGN NEW VARIABLE odd OF TYPE Boolean WITH true

FOR EACH perf OF TYPE Fund Performance IN **FUND's Year Performances**

```

    DO
      IF perf's YTD IS LOWER THAN 0
        THEN [perf's Title| perf's YTD-5| perf's YTD-4| perf's YTD-3| perf's YTD-2| perf's YTD-1| perf's YTD]
        ELSE
          IF odd
            THEN [perf's Title| perf's YTD-5| perf's YTD-4| perf's YTD-3| perf's YTD-2| perf's YTD-1| perf's YTD]
            ELSE [perf's Title| perf's YTD-5| perf's YTD-4| perf's YTD-3| perf's YTD-2| perf's YTD-1| perf's YTD]
      ASSIGN odd WITH NOT odd
  
```

**Cumulative Performance**

**Cumulative Performance Graph\***

**Country's Disclaimer\***

**Logotype**

# mbeddr 2014

```
double midnight2(int32 a, int32 b, int32 c) {
    -b + √(b2 -  $\sum_{i=1}^4 a * c$ )
    return  $\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a}$ ;
} midnight2 (function)
```

**Fig. 2.** Mathematical symbols used in C expressions embedded into C functions.

Events	
States	
beforeFlight	next(Trackpoint* tp)
airborne	[tp->alt > 0 m] -> airborne [tp->alt == 0 m && tp->speed == 0 mps] -> crashed [tp->alt == 0 m && tp->speed > 0 mps] -> landing [tp->speed > 200 mps && tp->alt == 0 m] -> airborne [tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne
landing	[tp->speed == 0 mps] -> landed [tp->speed > 0 mps] -> landing
landed	[ ] -> beforeFlight
crashed	[ ] -> beforeFlight

**Fig. 4.** A state machine represented as a table; also shows nested headers.

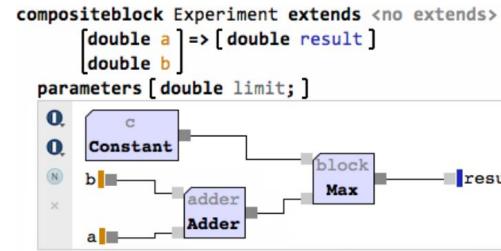
## 2.1 Hello, World

This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP [com.mbeddr.tutorial.zip](#) is available from the download page at [mbeddr.com](#). It is also part of the complete distro package.

**Fig. 6.** Margin cells used to support Word-like comments in MPS; other contents can be projected into the margin as well.

Here is a comment.  
23/06/14 15:58 (2 min ago) by markusvoelter

And a reply to it.  
23/06/14 15:58 (2 min ago) by markusvoelter



**Fig. 7.** A graphical editor embedded in a regular text editor.

## atomicblock Adder realizes IAdder

$[double a] \Rightarrow [double res]$   
 $[double b]$   
**contract**  $[pre(0) positive\_a: a > 0;$   
 $[pre(1) positive\_b: b > 0;$   
 $[post(0) sum: res == a + b;$   
 $ccode \{ res = a + b; \};$

$\cdot adps / \rightarrow des$   
 $\cdot adps / \rightarrow act$   
 $\cdot / \rightarrow desired$   
 $\cdot / \rightarrow actualI$   
 $\cdot / \rightarrow ambient$   
 $[double/Nms \rightarrow IntegralGain; ]$   
 $[double/Nm \rightarrow IntegralGain; ]$   
 $[double/Nm \rightarrow IntegralGain; ]$   
 $[double/Nm \rightarrow IntegralGain; ]$

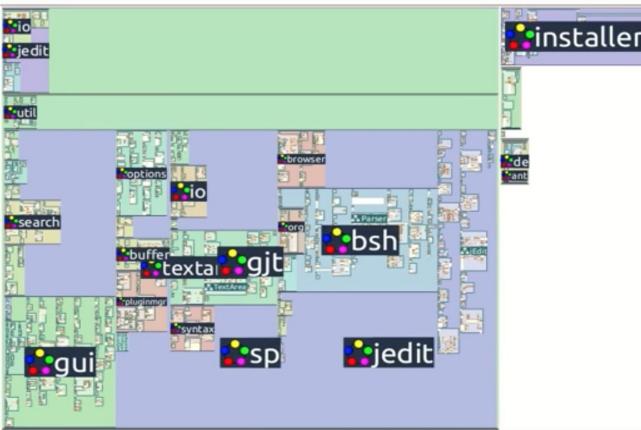
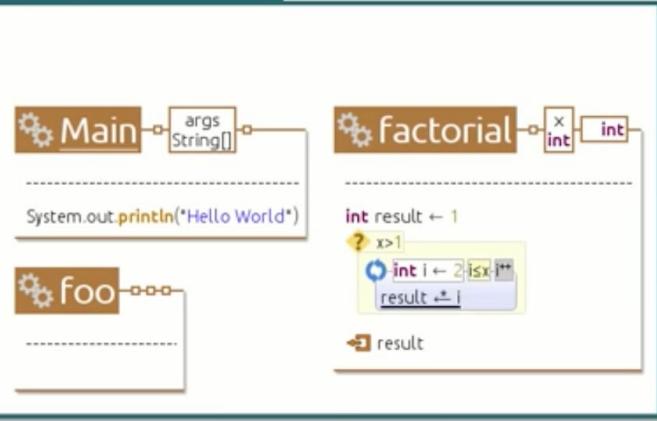
**Fig. 9.** A querylist is used to project the ports and contracts inherited from the interface realized by this block (in grey). New nodes ports or contracts can be entered above the grey lines.

**Fig. 10.** This tooltip shows the definition of the quantity referenced via the  $\rightarrow$  notation: it shows its type and various additional details. The tooltip uses the querylist to project derived nodes.

$$\text{result} = \frac{25300}{\left(\frac{20}{10 + 10 | \text{BASEPOINTS}}\right) * \left(\frac{1265}{1100 | \text{alt} + 165 | \text{speed}}\right)}$$

**Fig. 11.** This expression debugger renders the values of all subexpression over or to the left of the expression itself. The original expression (without the debug info) is  $(10 + \text{BASEPOINTS}) * (\text{alt} + \text{speed})$ .

# Hello



# Envision 2016

This screenshot provides a detailed view of the logic within the 'factorial' component. It uses `BigDecimal` for calculations, including multiplication and addition. It handles factor signs and carries, and includes exception handling for invalid input. The code is annotated with comments explaining the steps:

```
multiply(BigDecimal.valueOf(factor))
Duration.multiply(Factors.BigDecimal[])
BigDecimal carry ← ZERO
int FactorSign ← factor.signum()
factor ← factor.abs()

BigDecimal[] buf ← new BigDecimal [ 6 ]
int i ← 0 i<5 i++
BigDecimal bd ← getFieldAsBigDecimal(FIELDS[i])
bd ← bd.multiply(factor).add(carry)

buf[i] ← bd.setScale(0, BigDecimal.ROUND_DOWN)
bd ← bd.subtract(buf[i])
if i=1
    if bd.signum()≠0
        throw new IllegalStateException carry ← ZERO
    carry ← bd.multiply(FACTORS[i])

if seconds≠0
    buf[5] ← seconds.multiply(factor).add(carry) buf[5] ← carry

new DurationImpl

BigDecimal.getFieldAsBigDecimal(DatatypeConstants.Field)
```

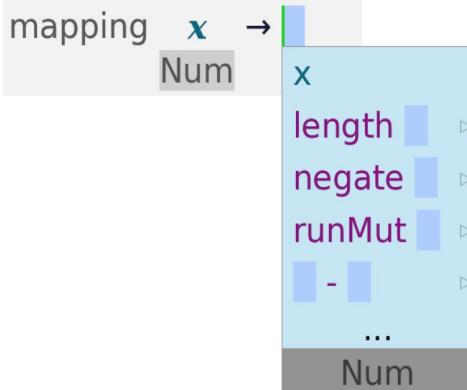
# Scheme Bricks 2008



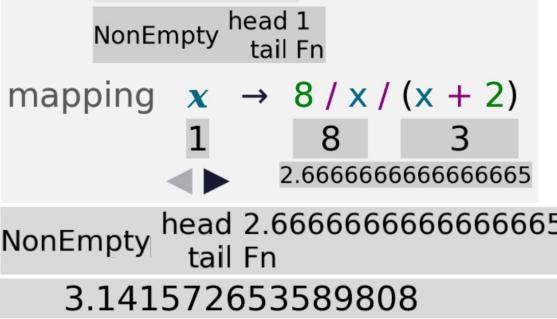
# Aardappel 2000



```
>>> sum map 1 .. 100000  
step 4
```



```
>>> sum map 1 .. 100000  
step 4
```



## Lamdu 2015

```
partition array cond =  
  Array a31 a31 → Bool  
  newMutArray « Empty »;  
    mutTrue →  
      MutArray Bl:val a31  
      s i67  
  newMutArray « Empty »;  
    mutFalse →  
      MutArray Bl:val a31  
      s i67  
sequence_ map fromArray array ;  
  mapping  $x \rightarrow$  appendMutArray cond x :  
    a31  
    True ↳ mutTrue  
    False ↳ mutFalse  
  val x  
freezeMutArray mutTrue ; trues → freezeMutArray mutFalse ; falses → return onTrue trues  
Array a31  
onFalse falses  
Array a31
```

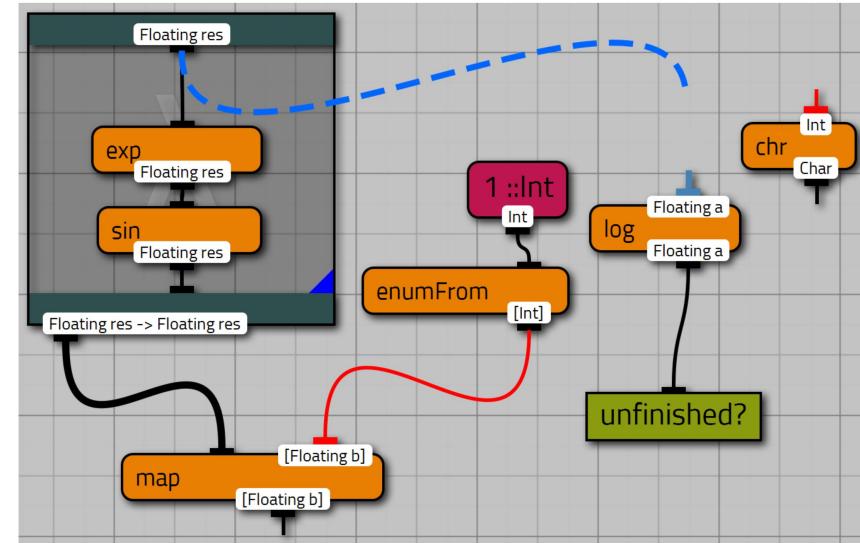
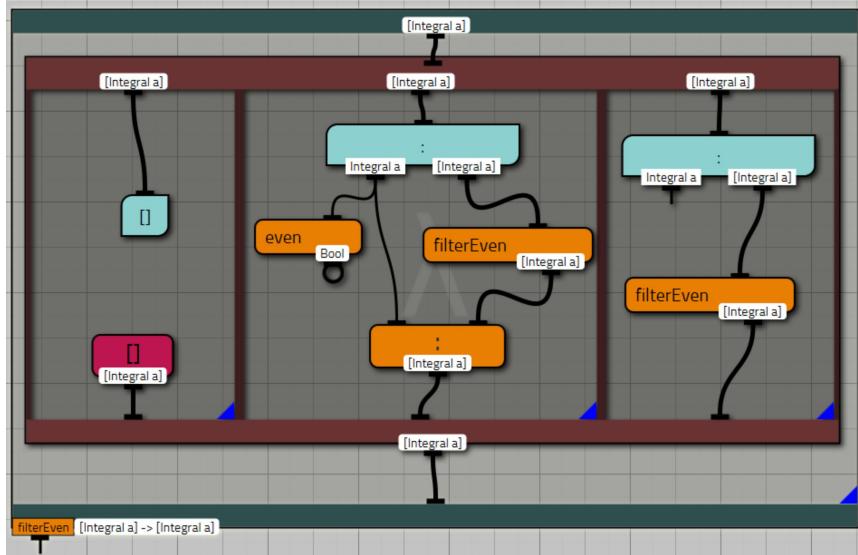
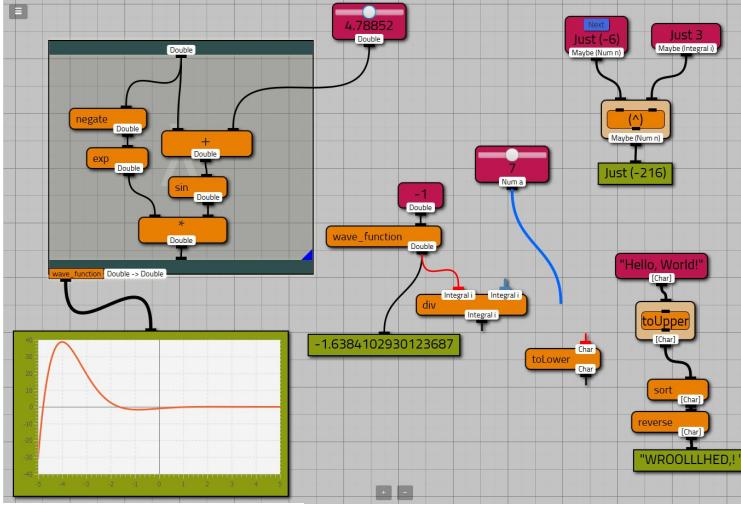
$(x == 0) \Rightarrow:$   
True ↳ "Hello,"  
False ↳ 5

Blame assignment

$(x == 0) \Rightarrow:$   
True ↳ "Hello,"  
Text  
False ↳ 5

# Viskell

## 2016



# Roly Perera

## 2012

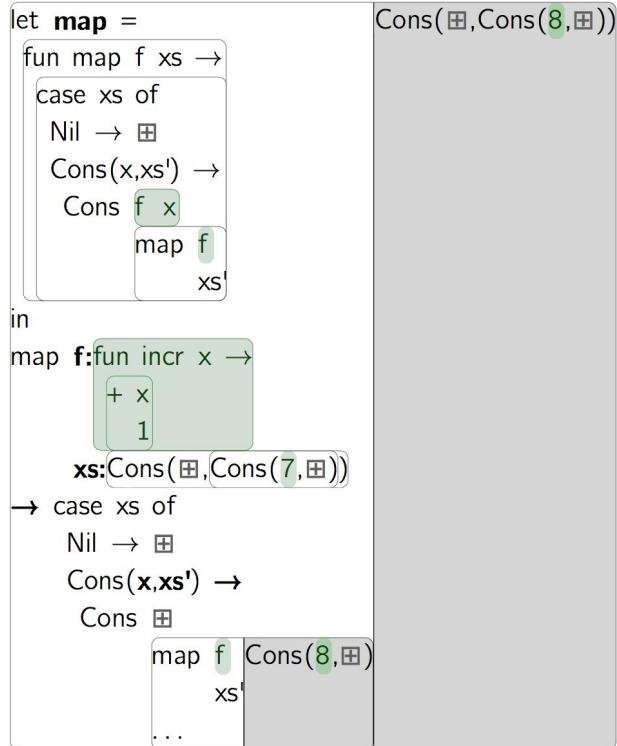
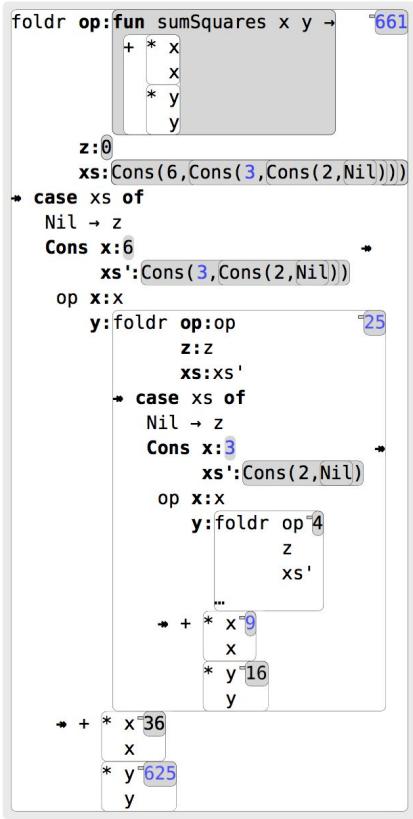


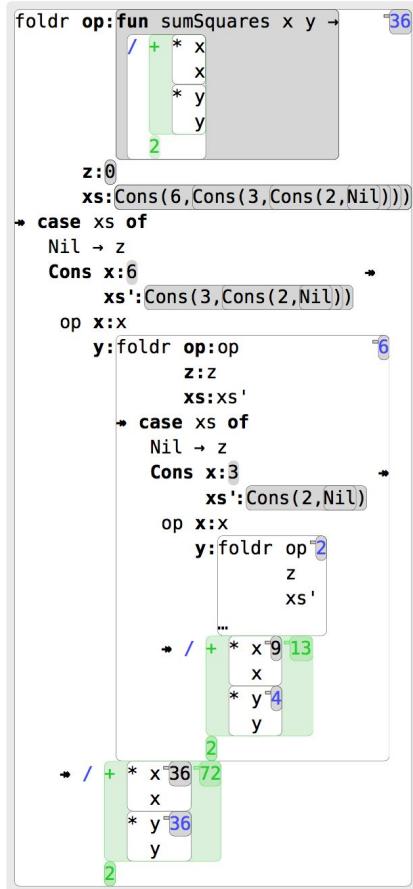
Figure 2. Slice of `map incr [6,7,2]`



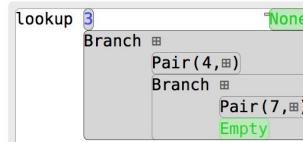
Figure 3. Outermost merge phase of mergesort.



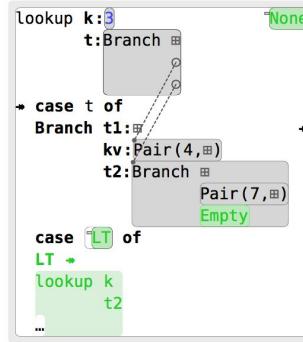
a. Expand nested op call



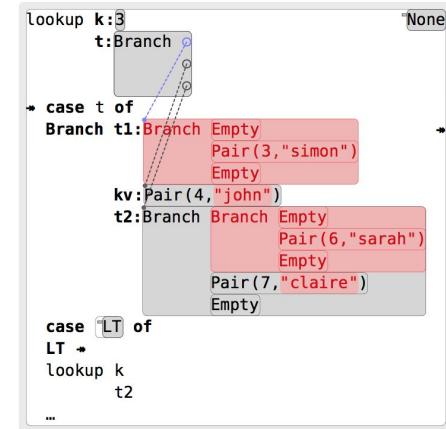
b. Change sumSquares



(a) Key 3 is not found



(b) Debug into test case

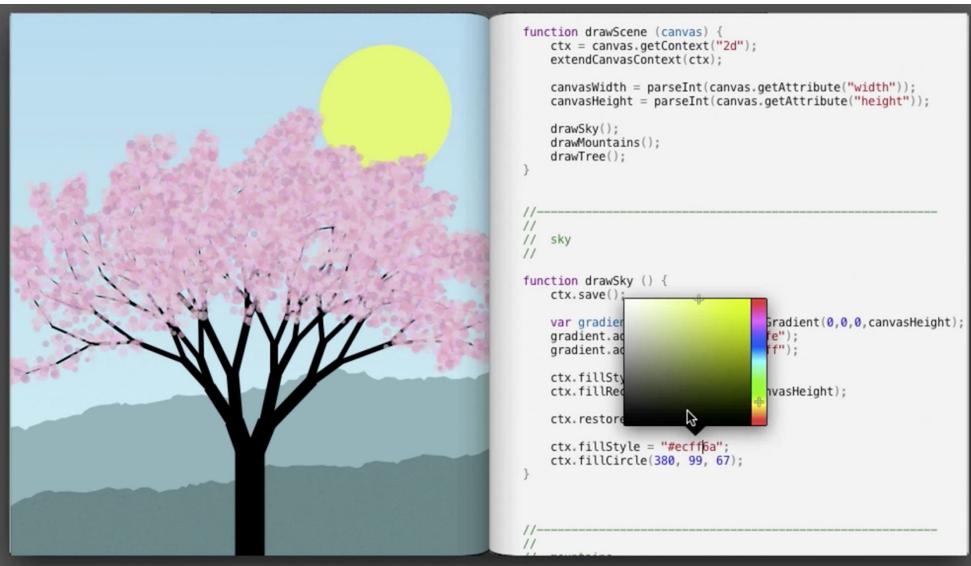


(c) Accept delta, then re-expand unused nodes

Figure 2.16 Finding and analysing bug in `lookup`

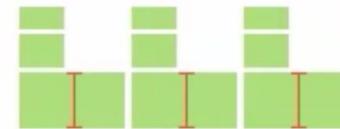
Figure 2.4 Structural editing with delta-highlighting

# Bret Victor: Inventing on Principle 2012



# Bret Victor: Learnable Programming 2012

```
fill(161, 219, 114);
for (var x = 40; x < 150; x += 50) {
    rect(x, 33, 20, 10);
    rect(x, 45, 20, 15);
    rect(x, 62, 47, 25);           height of the rectangle
}
```



```
var i = 0;
while (i < 20) {
    var scaleFactor = 1 + (20 - i)/20;
    resetMatrix();
    scale(scaleFactor);
    rotate(i * 6);
    fill(i * 30, i * 18, 0);
    triangle(0,0, 100,-20, 95,40);
    i += 1;
}
```



**Abstract**

var  
function

**Flow**

if  
for  
while

```
var x = 80;  
rect(x, 80, 40, 25);  
triangle(x, 80, 20, 60, 120, 80);
```

**Abstract**

var  
function

**Flow**

if  
for  
while

```
var x = 80;  
rect(x, 80, 40, 25);  
triangle(x, 80, 20 + x, 60, 120, 80);
```

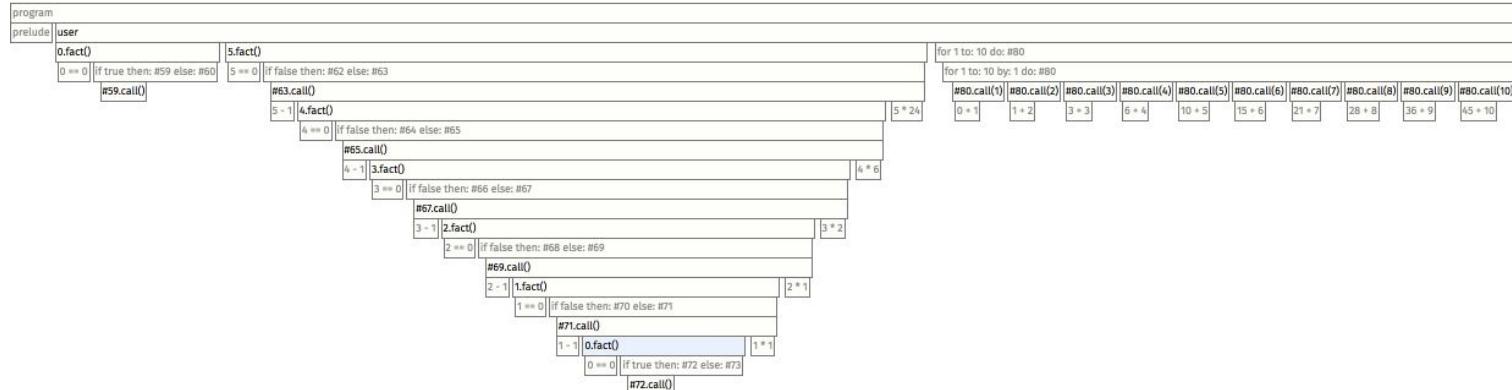
```
20 + x  
180 - x  
1.25 * x  
8000 / x
```



# Seymour 2017

```
1 def Number факт() { this = 0
2   if this == 0 then: {
3     return 1;
4   } else: {
5     var that = (this - 1).факт();
6     return this * that;
7   };
8 }
9
10 var f0 = 0.факт();
11 var f5 = 5.факт();
12
13 var sum = 0;
14 for 1 to: 10 do: {x |
15   sum = sum + x;
16 };
```

sum = 0  
x = 1 | x = 2 | x = 3 | x = 4 | x = 5 | x = 6 | x = 7 | x = 8 | x = 9 | x = 10  
sum = 1 | sum = 3 | sum = 6 | sum = 10 | sum = 15 | sum = 21 | sum = 28 | sum = 36 | sum = 45 | sum = 55



# Ohm Editor 2016

## Grammar

[local storage]

Save

Save As...

Saved

```
Arithmetic {  
    Exp = AddExp
```

```
    AddExp = AddExp "+" MulExp -- plus  
            | AddExp "-" MulExp -- minus  
            | MulExp
```

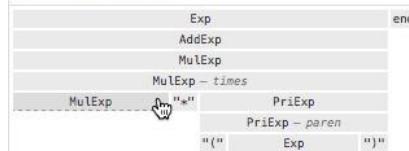
```
    MulExp = MulExp "*" PriExp -- times  
            | MulExp "/" PriExp -- divide  
            | PriExp
```

```
    PriExp = "(" Exp ")" -- paren  
            | "+" PriExp -- pos  
            | "-" PriExp -- neg  
            | number
```

```
    number (a number)  
        = digit* "." digit+ -- fract  
        | digit+
```

```
}
```

(1024 - 512) \* ( 256 + 8096 )



## Examples

Start rule: (default)



(1024 - 512) \* (256 + 8096)



x + 4

+ Add example

- Show example generator
- Explain parse
- Show spaces

# Subtext 1 2005

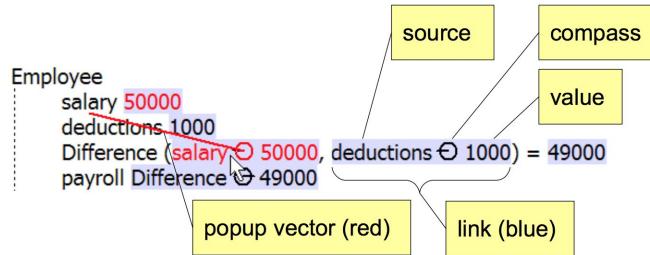


Figure 3. Linking

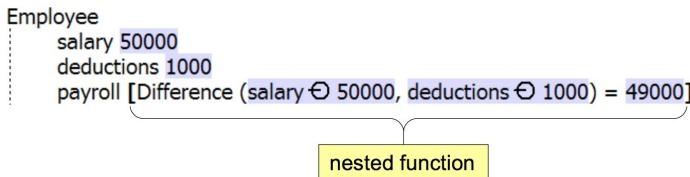


Figure 6. Nesting

Factorial *varies* Identity ⊖

first 1

Difference (first ⊖ 1, 1) = 0

Factorial *copies* Factorial ⊖

first Difference ⊖ 0

Difference (first ⊖ 0, 1) = -1

Factorial (Difference ⊖ 1) = Too deep!

Product (Factorial ⊖ Too deep!, first ⊖ 0) = Too deep!

Equality (first ⊖ 0, 0) = true

Choice (if Equality ⊖ true, then 1, else Product ⊖ Too deep!) = 1

= Choice ⊖ 1

Product (Factorial ⊖ 1, first ⊖ 1) = 1

Equality (first ⊖ 1, 0) = false

Choice (if Equality ⊖ false, then 1, else Product ⊖ 1) = 1

= Choice ⊖ 1

Increment *copies* Identity ⊖

first 0

= first ⊖ 0

Nest Sum within =

a) dragging to splice

Increment *copies* Identity ⊖

first 0

= [Sum (first ⊖ 0, 1) = 1]

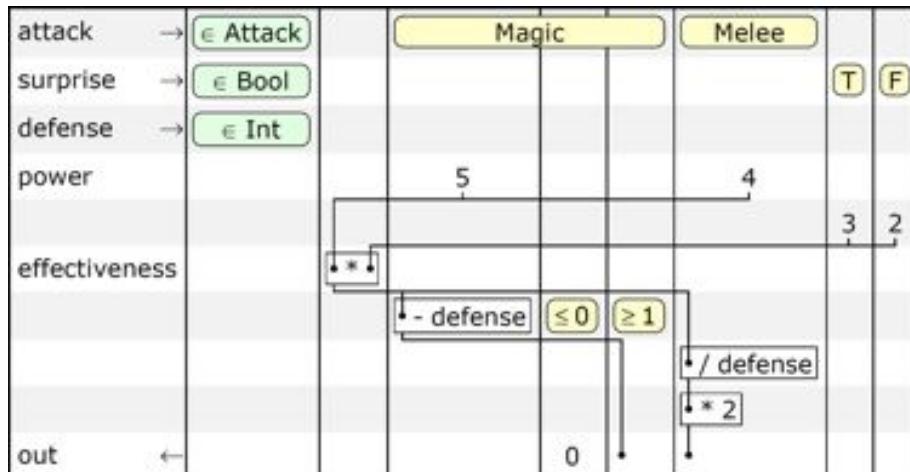
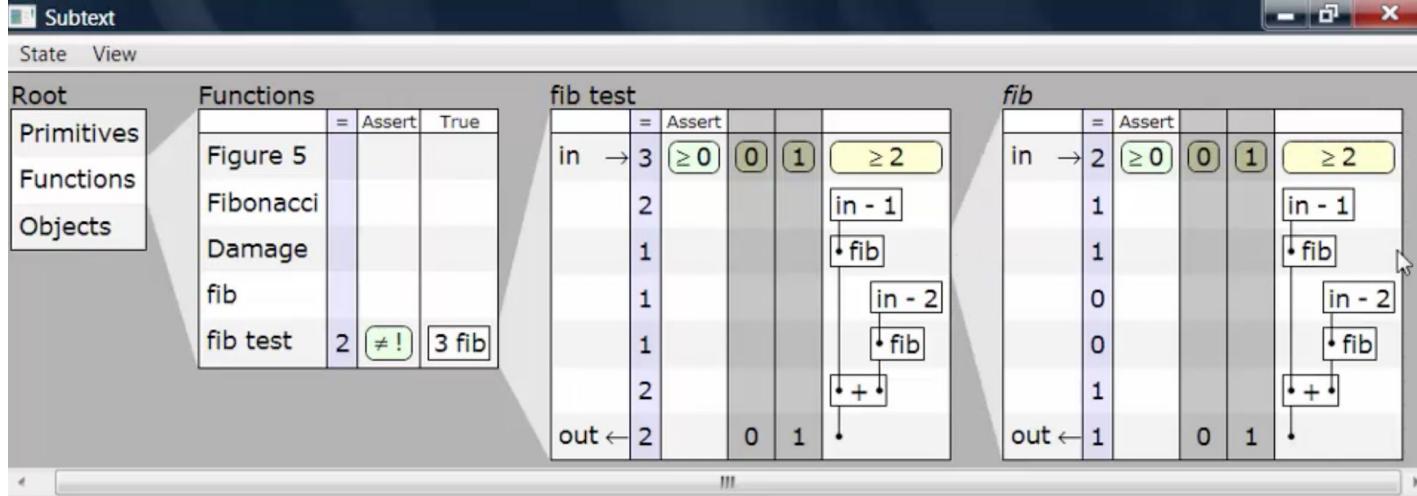
b) nested splice

Figure 7. Splicing

Figure 5. Factorial recursion

# Subtext 2 (Schematic Tables)

2007



## Subtext 5 (Two-way Dataflow) 2014

The screenshot shows a Subtext 5 interface with a toolbar at the top and a main workspace below. The workspace displays a two-way dataflow diagram for a Celsius to Fahrenheit conversion.

**Left Panel (Data):**

```
temperature: {
    celsius: 0
    fahrenheit = celsius * 9 / 5 + 32
}
UI = temperature
```

**Center Panel (Diagram):**

The diagram illustrates the computation of  $fahrenheit = celsius * 9 / 5 + 32$ . It starts with **celsius: 100** and branches into **number.\*** and **number./** operations. The **number.\*** path involves **celsius \* 9** (resulting in **900**) and then **900 / 5** (resulting in **180**). The **number./** path involves **celsius \* 9 / 5** (resulting in **180**) and then **180 + 32** (resulting in **212**). The final output is **fahrenheit = celsius \* 9 / 5 + 32** with a value of **212**.

**Right Panel (Changes):**

- celsius: -40** (Delta: **-40**)
- 40** (Delta: **-40**)
- celsius: -40** (Delta: **-40**)
- 40** (Delta: **-40**)
- 40** (Delta: **-360**)
- 360** (Delta: **-360**)
- 360** (Delta: **-72**)
- 72** (Delta: **-72**)
- 72** (Delta: **-40**)
- 40** (Delta: **-40**)

**Bottom Panel (Inputs/Outputs):**

celsius	100
fahrenheit	-40

# LogiX 2017

