4 Overview on Approaches to Multimedia Programming

Literature:

http://www.cs.sunysb.edu/~tony/364/historyofMM/historyofMM.html

Mark Guzdial: History of Squeak
Lecture notes at http://coweb.cc.gatech.edu/cs2340/3608

http://minnow.cc.gatech.edu/squeak/3139

Ivan Sutherland's Sketchpad, 1963

- First object-oriented drawing program
- Master and instance drawings
- Rubber bands
- Simple animations

Alan C. Kay

- U. Utah PhD student in 1966
  - Read Sketchpad, Ported Simula
- Saw “objects” as the future of computer science
- His dissertation:
  Flex, an object-oriented *personal* computer
  - A *personal* computer was a radical idea then
  - How radical?

"There is no reason anyone would want a computer in their home."
(Ken Olsen, Digital Equipment Corp, **1977**)

Further stations of Alan Kay’s life:
– Stanford Artificial Intelligence Laboratory
– **Xerox PARC**
– Apple, Atari
– Disney Interactive
– Viewpoints Research Institute
– Hewlett-Packard

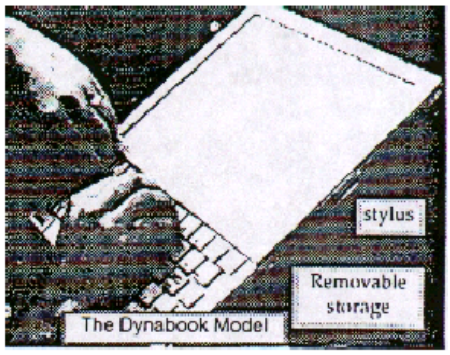
from M. Guzdial

The FLEX Machine

- “A personal computer for children of all ages”
- Includes pointing device

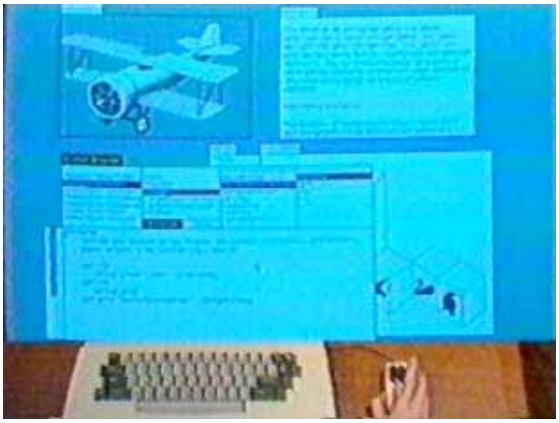
The Dynabook Vision

- Small, handheld, wireless(!) device – a new *medium*
- Can be used creatively by everybody, in particular children, for learning
- Xerox PARC Learning Research Group, early 70s



Tablet PC, 2004

Xerox PARC Learning Research Group: Smalltalk-72



- Object-oriented programming system
  - Mouse
  - Windows
  - Icons
  - Pop-up menus
- Uses simple object-oriented language "Smalltalk"
- Idea of user interface: Make computers easy to use for everybody
- Idea of language: make programming both more simple and more powerful (e.g. include *multimedia: sound*)

The Alto

- The machine the prototype of which impressed Steve Jobs so much that he decided to produce the Lisa/Macintosh kind of computers for the mass market (1979)
  - Graphical user interface
  - Networked via Ethernet
  - Programming language Smalltalk

History of Multimedia Authoring

- 1965: Douglas Engelbart: “Augmenting Human Intellect: A Conceptual Framework” (“intelligence amplifier”)
  - Leads to multiple windows, mouse, hypertext, composite text/graphic editing, outlining software, ...
- Text-based Hypertext Authoring:
  - 1965, Ted Nelson: “Hypertext”
  - 1968, Andries Van Dam: Hypertext Editing System
  - 1972, CMU: “ZOG” Hypertext collaboration tool
- 1976: DARPA proposal “multiple media”
- 1982, Peter Brown: “Guide”, Hypertext authoring with graphical interface
- 1984: Apple Macintosh
- 1985: Windows 1.0
- 1985: Commodore Amiga:
  - The first true multimedia computer (advanced graphics, sound, video)
- 1985: MIT Multimedia Lab (Negroponte, Wiesner)
- 1986: Xerox PARC “NoteCard”
- 1987: Apple HyperCard
- 1988: Macromedia Director

Commodore Amiga

- Erscheinungsjahr: Mitte 1985 (Deutschland 1986)
- Arbeitsspeicher 256KByte Chip-Ram
- CPU: Motorola 68000, 7,16Mhz NTSC, 7,09Mhz PAL
- Grafik (u.a.):
  320*200/256 (32/4096 Farben)
  640*200/256 (16 Farben)
- Sound: 8Bit 4 Kanal Stereo, 29Khz
- Massenspeicher: 1 Diskettenlaufwerk 880KByte
- Betriebssystem: Kickstart 1.0, 1.1, 1.2, 1.3 (Rom auf Diskette)
- Einführungspreis: 6000,- DM

Atari Mega ST

- 1988
- CPU Motorola 68000 8 MHz
- Ram 1 bis 4 MByte
- Grafik
  - 640 x 200 (4 Farben)
  - 320 x 200 (16 Farben)
- Tongenerator: 3 Stimmen
  - MIDI Interface
- Ca. 2000 DM
- Typische Anwendungen:
  - Spiele
  - Musik
  - Ausbildung (Schulen)

4 Overview on Approaches to Multimedia Programming

Literature:

http://www.squeakland.org

Back to the Future: Squeak

- Smalltalk:
  - Developed 1972
  - Commercial versions from 1980 on
- 1995: Alan Kay, Dan Ingalls, Ted Kaehler at Apple
  - Still want "A development environment in which to build educational software that could be used—and even programmed—by non-technical people and by children"
  - Build on Open Source Software strengths
    - » Use the distributed power of Internet-based programmers
  - Available Smalltalk versions had lost many media capabilities
- Later on, the Squeak team moves to Disney
  - “Its all about media”
- Multimedia in Squeak:
  - 16 voice music synthesis
  - 3-D graphics, MIDI, Flash, sound recording
  - Network: Web, POP/SMTP, zip compression/decompress

Squeak as a Classroom Tool for Schools

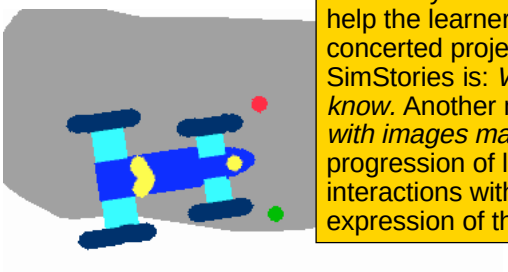
- Vision:
  - Children use complex multimedia computations (graphics, sound, animations) like a desktop calculator
- Example: Physics
  - experiments regarding physical observations
    - » Building a computer model of real-life behaviour
  - Video: The Gravity project
- Prerequisites:
  - Fully visual programming
  - Large coverage of graphics and animation

4 Overview on Approaches to Multimedia Programming

Literature:
http://www.squeakland.org

Electronic Toys: EToys

Alan Kay, EToys and Simstories in Squeak



"EToys are computer environments that help people learn ideas by building and playing around with them. They help an "omniuser" - usually a child - create a satisfying and enjoyable computer model of the idea and give hints for how the idea can be expanded. SimStories are longer versions of EToys that - like essays - string several ideas together to help the learner produce a deeper and more concerted project. A good motto for EToys and SimStories is: *We make not just to have but to know.* Another motto that applies here is: *Doing with images makes symbols.* That is, the progression of learning moved from kinaesthetic interactions with dynamic images to a symbolic expression of the idea."

Basics of Squeak Interaction (1)

- Squeak assumes a three-button mouse
- Menus are invoked by clicking on objects
  - clicking on surface opens "world" menus
- "Red"
  - Windows: left-button click
  - MacOS: simple click
- "Yellow"
  - Windows: middle-button click
  - MacOS: option + click
- "Blue"
  - Windows: right-button click
  - MacOS: + click

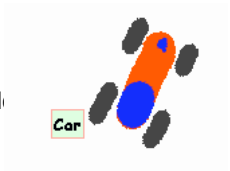
(A different colour mapping...)

Basics of Squeak Interaction (2)

- Flaps:
  - Areas which can be opened or closed in a drawer-style
  - Often used as repositories (“parts-bins”)

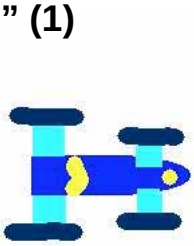
- Collapsing windows:
  - A window can be collapsed or expanded

- Tiles:
  - Objects can be represented by “tiles” (see bel



Etoys: Example “Car Race” (1)

- Step 0:
  Create a new empty project
  - world menu -> open...
    -> morphic project
  - enter new project by double-click
- Step 1: Draw the things with which we want to play
  - Very simplistic bitmap-oriented painting tool
- Step 2: “Keep” the drawing
  - We get a Squeak object
    - » Free form, not square
  - Can be moved around

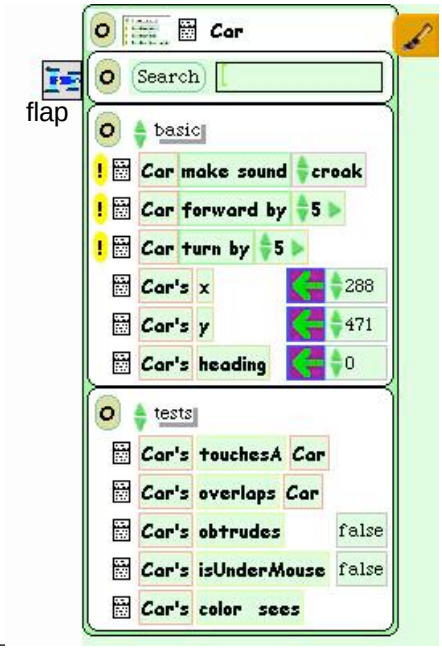


“Halo” of a Squeak Object

- The “halo” is a circular graphic menu which can be invoked on any object by a mouse click
  - “blue” click
  - special “playfield configuration” (preferences): invoked just by mouse over

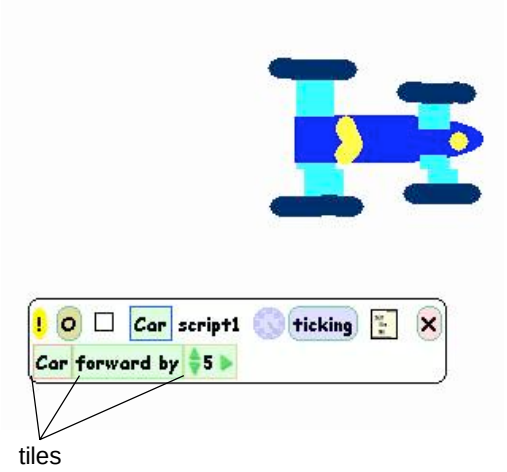
Squeak Viewers

- Step 3: Create a viewer (e.g. via the object’s halo)
  - Special flap for quickly showing and hiding the viewer
  - Rename sketch in viewer e.g. to “Car”
- Shows categories of properties and commands for objects
  - Categories: Object is derived from a subclass in a complex class hierarchy
  - Viewer can show many different categories in parallel
- Commands can be immediately executed (exclamation mark button)
  - Car can be moved, turned (Note: Orientation to be set in “rotate” mode to define direction of movement)



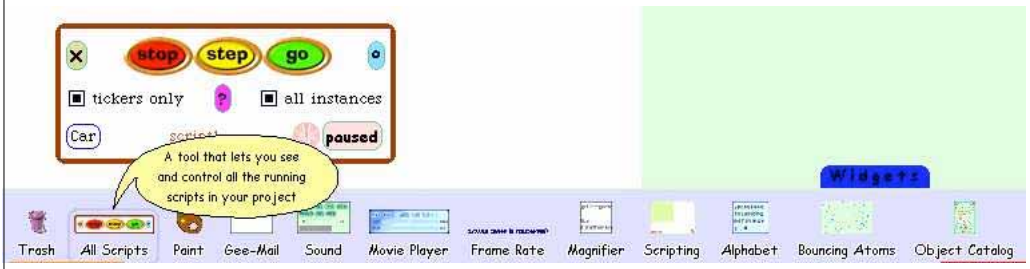
Squeak Scripts

- *Script*:
  - simple sequence of commands
  - executed under user control or automatically through a timer ("ticking").
- Represented by windows
  - created by drag-and-drop
  - "Tiles" represent objects and actions
- Step 4: Create a script
  - "add new script" in viewer
  - drag "empty script" onto surface
- Step 5: Add forward command
  - drag it from the Car viewer
  - adjust the parameter(s)



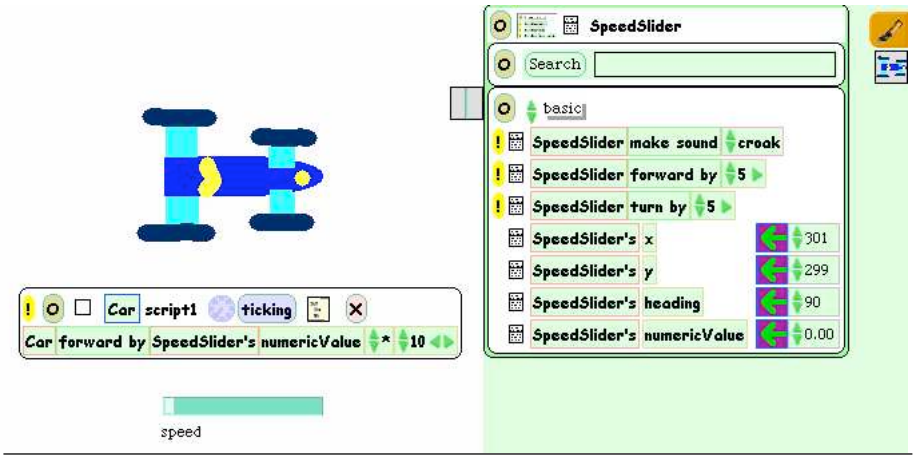
Running a Script

- Step 6: To control all scripts, use a new script control object.
  - To be found under the "Widgets" flap, like many other helpful tools
- All scripts of the project are simultaneously started and stopped through one button
  - Again just one drag operation to instantiate the object
- Example: Now car can be "driven" forward (till the border of the screen)



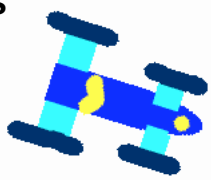
Object Interaction in Scripts

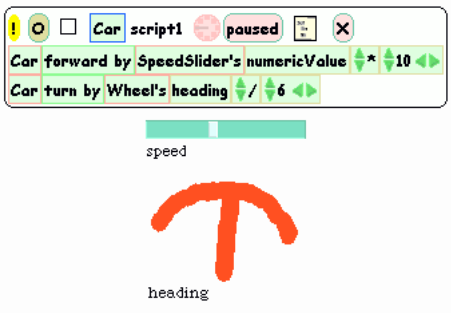
- Parameters of script commands can be computed from other objects' properties (by dragging the property onto the parameter location)
- Local adjustments can be added at the end (factor, offset etc.)



User Control through Graphical Objects

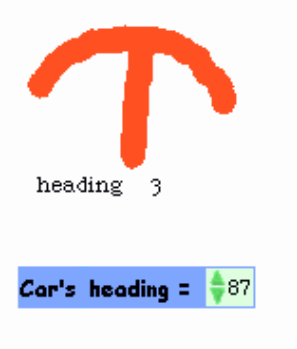
- Graphical manipulations can be used to control other objects
- Example:
  - Steering wheel graphics
    - » Drawn by hand
    - » Viewer attached
  - Rotated by user (e.g. through halo operations)
  - Heading of wheel is transferred to car
  - A "servo steering" i.e. a less sensitive transfer is recommendable





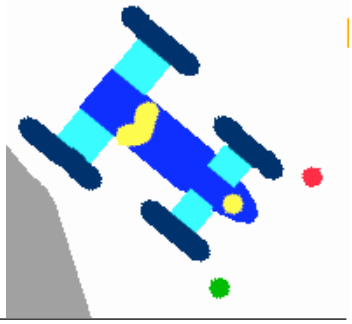
Watcher

- The values of object properties can be easily shown on the screen
  - Updated regularly and automatically
- Technically, this is an “Observer” mechanism
  - Hidden behind simple drag&drop interface
- Watcher:
  - Simple watcher (value), Detailed watcher (value plus label)
  - Can be obtained from menu left of property (in viewer)
  - Can be placed anywhere on screen

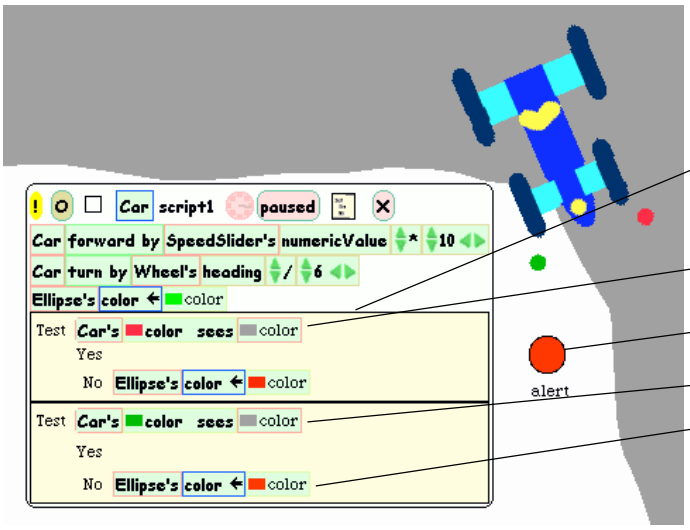


Sensors for Environment

- Squeak objects can easily observe where they are currently located
  - Through coordinates
  - Simpler: through colours
- *Sensors*:
  - Realizable as special parts of the graphics with a unique colour
  - “color x sees color y” test: Which colour is below the sensor?
- Example:
  - Grey road, car with two sensors
  - Alert lamp shall go red when one of the sensors is not on road



Example: Alert Lamp



Test tile

Test on left sensor

Alert lamp

Test on right sensor

Assignment (dragging the green-on-purple arrow right of properties)

Example: Auto-Steering

- Interaction among objects can be designed in communication loops
- Example:
  - Car automatically moves forward
  - Sensor detects border of road
  - Car automatically steers to stay on the road
- Enables complex interactive learning experiences (setting up feedback loops)

Wheel control better removed at this stage?