Newton Introduction

Jeffrey C. Schlimmer
School of EE & CS
Washington State U

20 Oct 93

Newton Technology

What is Newton Technology?

- Project started at Apple in 1988
- Goal: Bring computing power to the masses
- Result:
  - Assist people in their day-to-day tasks
  - Accessible and easy to use

What People Do:

- Capture
  - Jot, sketch, scribble, figure, doodle, takes notes, make lists …
- Organize
  - Clean up, find, format, schedule …
- Communicate
  - Send, share, route, print, fax, mail …

Newton Technology

- Hardware
  - Focus on mobility & communications
- Software
  - Handwriting recognition
  - [Dog] Intelligent assistance
- Applications
  - Common pool of data

Newton is Not a Small Computer (?)

Newton	*Small Computers*
Focused function	General purpose
New architecture	Desktop architecture
New applications	Existing applications
Personal OS	Multi-user OS

The Newton MessagePad

"The first in a family of products"

- PCMCIA Card Slot
- Infrared Beam
- Communication Port
- Speaker
- Touch Sensitive Screen
- Sleep Switch

Newton Licensing

"The first in a family of products"

- Technology licensed to others
  - Sharp, Siemens/ROLM, Motorola, Matsushita, Alcatel, Ameritech, Arm, Bell South, British Telecom, Cirrus Logic, Deutsche Telekom, GEC Plessey Semiconductors, LSI Logic, US West
- Future versions likely to be different size and function

Newton Operating System

- Based on persistent objects
- Object-oriented
- Open, inspectable user data
- Designed for extension
  - Extend applications
  - Extend hardware

Persistent Objects

- All user data is stored in objects
  - Objects are always in memory
  - Objects can be beamed, faxed, …
  - Objects can be synchronized
- All applications read/write to objects

Newton Memory

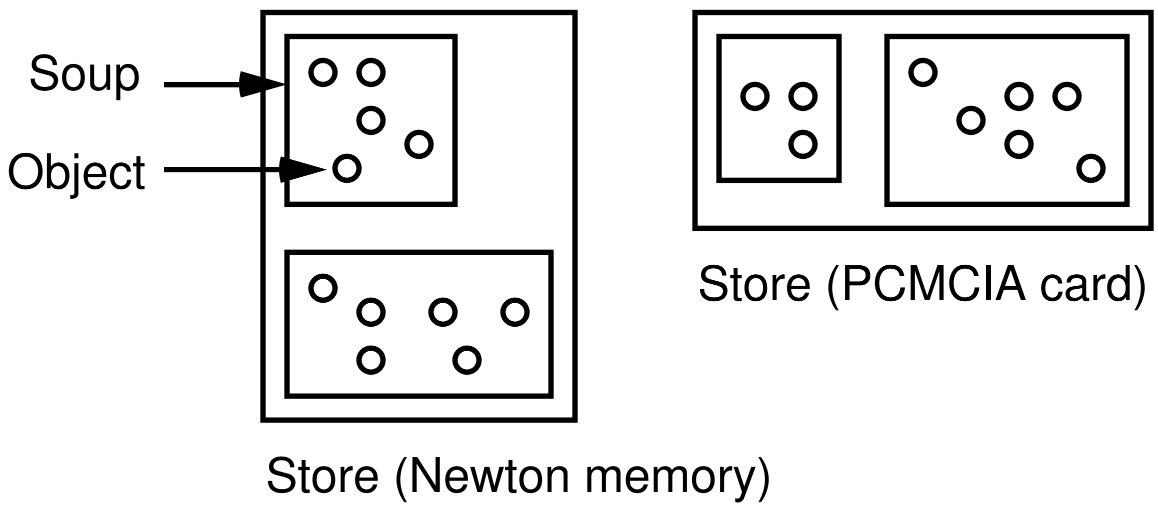
- 4 MB of ROM in current MessagePad
- 640K RAM (192K usable)
- Extendable via PCMCIA memory cards
  - Combination of ROM, RAM, & electronics
  - Some FAX cards available

Newton Operating System

- Significant emphasis on ROM
  - Cheaper
  - More reliable
  - Less power
- What goes in ROM?
  - System defined classes
  - Programmer defined classes

Object Store Concepts

- Stores – physical locations
- Soups – collections of related objects



Newton Soups

- Calendar application
  - Calendar, Calendar Notes, Repeat Meetings, Todo
- Names application
  - Directory, Names, Notes, Repeat Notes
- General
  - Inbox, Outbox, System

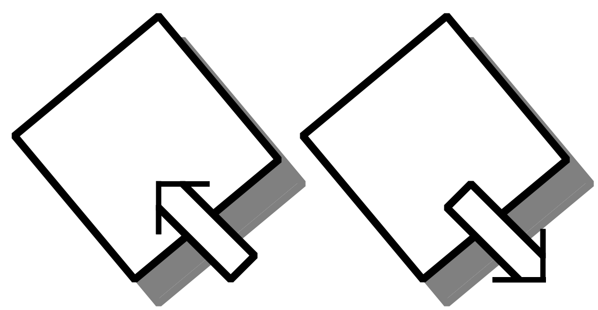
Intelligent Assistance

"Intelligent" is a buzzword

- IA provides another means for users to do what they want to do
- Use language instead of finding and launching
  - E.g., "Call Bob"
- Built-in support for find, todo, mail, fax, printing, calling, & time/location info

Routing and Communication

- Every application is communication aware
  - Print, mail, fax, beam
- A single In box
- A single Out box
- Store and Forward



Finding

- Accesses one or all soups
- Searches for string
  - Specified by user
  - Needed for a confirmation slip
- All applications can use find
- All soups are find-able

Newton Programming Language

NewtonScript

- Goals
  - Object-oriented
  - Portable
  - Machine independent
  - Extremely efficient use of RAM

Why a New Language?

- Object-oriented dynamic language
  - Garbage collection
  - Introspection (query self, query existence)
  - Run-time typing
- Efficient use of RAM

NewtonScript

- Based on Lisp
  - Dynamic memory
  - Dynamic typing
  - Every function returns a value
- Strong distinction
  - Class (ROM)
  - Instance (RAM)

NewtonScript Examples

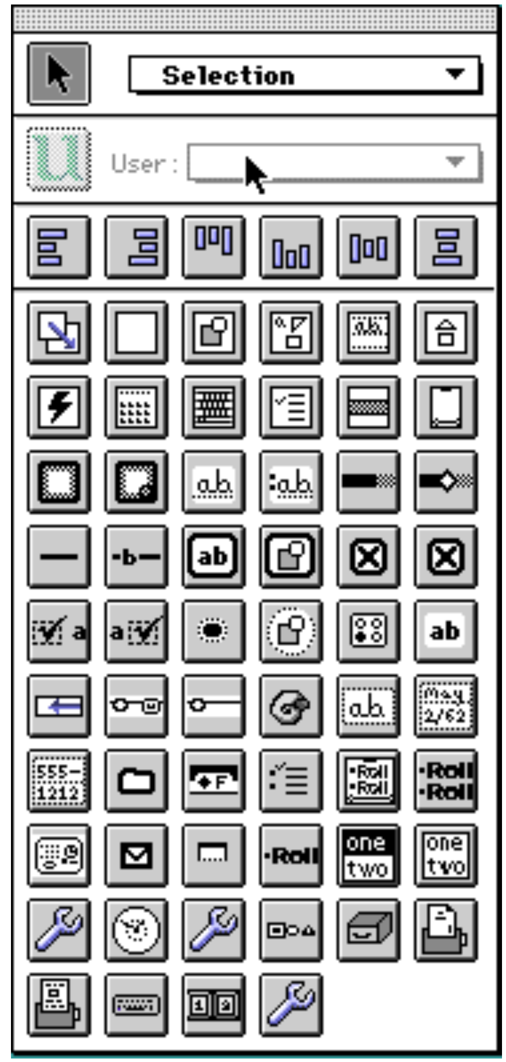
```
• myFunctions := {                        // assign a new frame
    factorial: func (n)                   // a slot called FACTORIAL which
    begin                                 // is a function.
      if n ≤ 1 then                       // check for base case
        return 1                          // just return 1
      else                                // otherwise
        return n* :factorial(n-1);        // recurse.
    end,                                  // end of function
    factorial2: func (n)                  // second slot in frame which
    begin                                 // is a function.
      local total := 1;                   // local variable
      for i := 2 to n do                  // iteration
        total := i * total;               // to compute n!
      return total;                       // value of function
    end,                                  // end of function
  };                                      // end of new frame
```

NewtonScript Examples

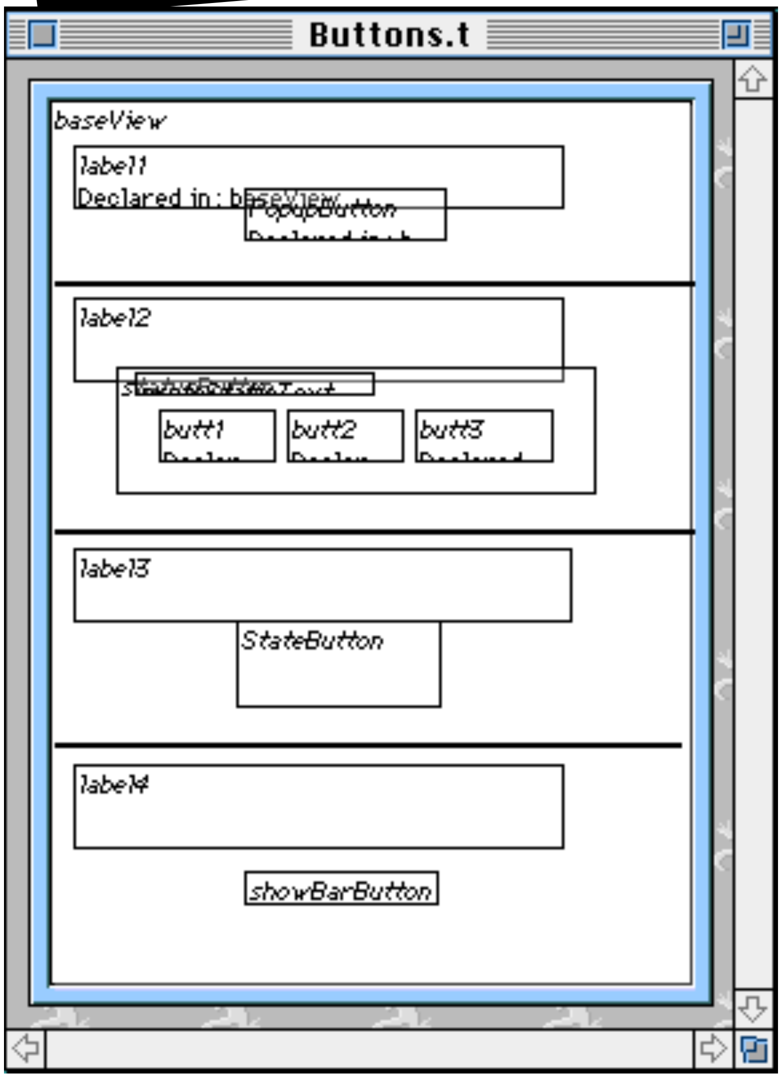
- Frames
  - Basic NewtonScript data structure
  - Like records or structs–but dynamic
  - Have slots with values
    - {slot1: 29, slot2: {slot3: "foo"}}
- Objects in NewtonScript are frames
  - Stored in soups (RAM)
  - Stored in classes (ROM)

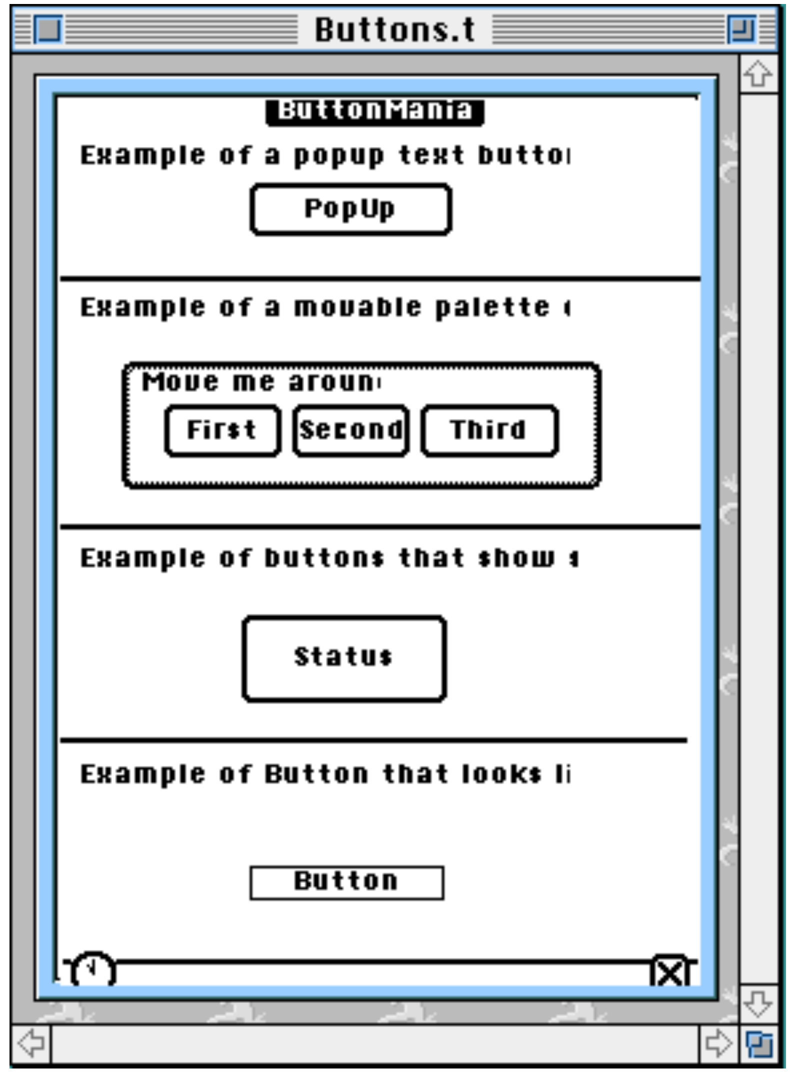
Visual Classes

- Every class has a visual representation (Even user defined classes)



Visual Programming Environment





Outlook for Newton OS

- Significant departure from protected information
- OS hooks and user expectation for intelligence
- Highly embedded computing