RUBY: Reminiscing about User interfaces by Brad over the Years

Brad A. Myers
Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
http://www.cs.cmu.edu/~bam
bam@cs.cmu.edu

© 2017 – Brad A. Myers
40th Anniversary!

**RUBY**
**R**eminiscing about
**U**ser interfaces by
**B**rad over the
**Y**ears
35 CHI conferences: 1982 - 2017

© 2017 – Brad A. Myers
Lots of Badges
35 CHI conferences: 1982 - 2017
Conference ribbons

1990

(Thanks to Ben Shneiderman for the picture)
Baby me
Nerdy me
~1974 - Teletype

5 LET S = 0
10 MAT INPUT V
20 LET N = NUM
30 IF N = 0 THEN 99
40 FOR I = 1 TO N
50 LET S = S + V(I)
60 NEXT I
70 PRINT S/N
80 GO TO 5
99 END

Observations

• Brad is really old…

• If you don’t keep stuff, you won’t have it 40 years later!
Spatial Data Management System (SDMS)

• ~ 1977
• MIT Architecture Machine Group
  • Predecessor of MIT Media Lab
• Undergraduate research

Xerox PARC

- Important research lab
- Summer intern

Picture credit: http://www.digibarn.com/collections/locations/xerox-parc/
Smalltalk group at PARC

• Summer 1977
• Systems Sciences Lab (SSL)
• Alan Kay
• Also: Dan Ingalls, Adele Goldberg, Ted Kaehler, Bruce Horn, etc.
Computer Sciences Lab (CSL) at PARC

- Summer 1978
- Summer & Fall 1979 – Master’s Thesis
- Used the “Mesa” programming language
- Advised by: Dan Swinehart (PARC) and David Reed (MIT)
- Helped by: John Warnock, Ed Satterthwaite, Butler Lampson, Warren Teitelman, Bill Paxton and Paul Rovner
Incense

- Displays data structures similar to how they might be drawn on a blackboard
- Customizable
INCENSE:
A SYSTEM FOR DISPLAYING DATA STRUCTURES

Brad A. Myers*  
Xerox Palo Alto Research Center, California
Observations

• It’s better not to write out and read a talk.
Three Rivers Computer Corporation

- 1980-1983
- Pittsburgh, PA
- PERQ workstation
Bernita

• 1982
Bernita

- 1982
- 35 years!
Sapphire for PERQ

- 1983
- SAPPHIRE Screen Allocation Package Providing Helpful Icons and Rectangular Environments

© 2017 – Brad A. Myers
Percent-Done Progress Indicators

- CHI’1985 paper

Static busy signals

Progress indicators are better
Two-Handed Input
[Buxton and Myers, CHI’1986]
Taxonomy, 1986

- Programming by Example vs. with Example
- Visual Programming vs. Program Visualization

Visual Programming, Programming by Example, and Program Visualization: A Taxonomy.

Brad A. Myers
Dynamic Graphics Project
Computer Systems Research Institute
University of Toronto
Toronto, Ontario, M5S 1A4
Canada

1. Introduction

As the distribution of personal computers and the more powerful personal workstations grows, the majority of common users now do not know how to program. They
Observations

• A good taxonomy or survey can be a nice contribution
Peridot, 1986-87

- **PERIDOT**
  Programming by Example for Real-time Interface Design Obviating Typing

```
Parameters to scrollbar
CharsInFile = 5036
Active Values:
**Mouse** - (320 120 NIL T NIL
ScrollPercent ~ 20
WhereInFile ~ 3521
```

(a) (b) (c) (d) (e) (f)

CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=996283
Peridot, 1986-87

Editor Function

Cut
Paste
Search
Read File
Save File
Format
Paginate
Looks

Bold
Italic
Underline
Inverted
Strike Through
Shadow
Outline

18 %

Slide To Select Grey Shade:
Observations

• If you are first, then you can just show “it can be done”
• To show “better” requires different evidence
  • User study or measurements
• Showing it can be used by certain users
• Depends on what you want to claim
Observations

- Having kids while a graduate student can work out since you have more flexible time than when you are a professor.

Ryan Myers was born in Toronto, Canada in August, 1985.
My Four Sons, 1993
Family Today

• 2016
Carnegie Mellon University, 1987…

• 30 years this summer!
Garnet, 1988 - 1994

- **GARNET**
  Generariong an Amalgam of Real-time, Novel Editors and Toolkits
Prototype-Instance Object System

- Any object can be the *prototype* for other objects
- Now used by JavaScript
- Changes to prototype propagate
- *Structural inheritance*
Constraints

• Values automatically computed based on dependencies
• Connect graphics
• Showed useful for **copying values**
  • “Data bindings”
• Introduced **“pointer variables”** for constraints
  • E.g., “same size as whatever is selected”
Retained Object Model

- “Display list” for graphics
- Automatically handles refresh
- Like DOM for web pages today
“Interactors” Input Model

• Parameterized objects for each kind of behavior
• Key behaviors used in GUIs
• Independent of the graphics
• Reusable for all interactions
• Never need to write event handlers
• Concept used by Adobe Flash Catalyst, etc.

—Menu-Interactor—select,
—Move-Grow-Interactor—position,
—New-Point-Interactor—position,
—Angle-Interactor—orient,
—Text-Interactor—text,
—Trace-Interactor—path.
### Higher Level Tools in Garnet

<table>
<thead>
<tr>
<th>LAPIIDARY</th>
<th>AGATE</th>
<th>GILT</th>
<th>MARQUISE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisp-based Assistant for Prototyping Interface Designs Allowing Remarkable Yield</td>
<td>Gesture-recognizer And Trainer by Example</td>
<td>Graphical Interface Layout Tool</td>
<td>Mostly Automated, Remarkably Quick User Interface Software Environment</td>
</tr>
</tbody>
</table>

© 2017 – Brad A. Myers
- **CHI’1991**

<table>
<thead>
<tr>
<th>Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>10</td>
</tr>
<tr>
<td>Top</td>
<td>10</td>
</tr>
<tr>
<td>Width</td>
<td>50</td>
</tr>
<tr>
<td>Height</td>
<td>50</td>
</tr>
<tr>
<td>Visible</td>
<td>T</td>
</tr>
<tr>
<td>Line-Style</td>
<td>OPAL:DEFAULT-L ...</td>
</tr>
<tr>
<td>Filling-Style</td>
<td>NIL</td>
</tr>
<tr>
<td>Draw-Function</td>
<td>COPY</td>
</tr>
<tr>
<td>Window</td>
<td>W</td>
</tr>
<tr>
<td>Parent</td>
<td>A</td>
</tr>
<tr>
<td>Is-A</td>
<td>OPAL:RECTANGLE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>&quot;C32&quot;</td>
</tr>
<tr>
<td>Font</td>
<td>OPAL:DEFAULT-F ...</td>
</tr>
<tr>
<td>Left</td>
<td>25</td>
</tr>
<tr>
<td>Top</td>
<td>28</td>
</tr>
<tr>
<td>Width</td>
<td>21</td>
</tr>
<tr>
<td>Height</td>
<td>14</td>
</tr>
<tr>
<td>Visible</td>
<td>T</td>
</tr>
<tr>
<td>Line-Style</td>
<td>OPAL:DEFAULT-L ...</td>
</tr>
<tr>
<td>Fill-Backgrou</td>
<td>NIL</td>
</tr>
<tr>
<td>Actual-Height</td>
<td>NIL</td>
</tr>
<tr>
<td>Draw-Function</td>
<td>COPY</td>
</tr>
<tr>
<td>Window</td>
<td>W</td>
</tr>
</tbody>
</table>

© 2017 – Brad A. Myers
Silk

- James Landay PhD, 1996

- **SILK** Sketching Interfaces Like Krazy
3 Generations of CHI Academy

• Baecker, Buxton, Myers, Landay
<table>
<thead>
<tr>
<th>TOURMALINE</th>
<th>GAMUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text-formatting</td>
<td>Games</td>
</tr>
<tr>
<td>Ought to</td>
<td>Are</td>
</tr>
<tr>
<td>Use and</td>
<td>Made</td>
</tr>
<tr>
<td>Rely on</td>
<td>Using</td>
</tr>
<tr>
<td>Macrostyles</td>
<td>This</td>
</tr>
<tr>
<td>And</td>
<td></td>
</tr>
<tr>
<td>Layout</td>
<td></td>
</tr>
<tr>
<td>Inferred</td>
<td></td>
</tr>
<tr>
<td>Nicely by</td>
<td></td>
</tr>
<tr>
<td>Example</td>
<td></td>
</tr>
</tbody>
</table>
TOURMALINE
Text-formatting
Ought to
Use and
Rely on
Macrostyles
And
Layout
Inferred
Nicely by
Example

GAMUT
Games
Are
Made
Using
This

GOLD
Graphs and
Output
Laid-out by
Demonstration
All the Widgets, CHI’1990

- Scroll Bars
- Menus
- Palettes
- Command Buttons
- Radio Buttons
- Check Boxes
- Text Selection
- Basic Text Editing
- Dialog Boxes
- Selecting Graphics
- Basic Editing of Graphics
- Desktops
- Window Manager
- Commands

- **AMULET**
  Automatic Manufacture of Usable and Learnable Editors and Toolkits
Command Objects

• **TOPAZ**
  Transcript
  Of
  Programs
  Activated with
  Zeal
Pebbles, 1998 -

- PEBBLES
  PDAs for
  Entry of
  Both
  Bytes and
  Locations from
  External
  Sources
Pebbles Projects

• Command Post of the Future
  • Successful DARPA program
  • Expected to be about speech/gestures
  • Actual issues were mainly situation awareness
    • Including connected mobile devices
Personal Universal Controller (PUC)

- Jeff Nichols PhD, 2006
Personal Universal Controller (PUC)
EdgeWrite

- Jake Wobbrock, PhD, 2006
Sugilite

• Current PhD work of Toby Li
  • See our paper, Thurs, 11:30 - 12:50 in Room 203

• SUGILITE
  (come to the talk to see what it stands for!)
Natural Programming, 1995-

- Make programming easier and more correct by making it more natural
- Closer to the way that people think about algorithms and solving their tasks
- Novice, expert, and “end-user” programmers

[Laforenza, Shaw and Myers 2005]
“Millions for compilers but hardly a penny for understanding human programming language use. Now, programming languages are obviously symmetrical, the computer on one side, the programmer on the other. In an appropriate science of computer languages, one would expect that half the effort would be on the computer side, understanding how to translate the languages into executable form, and half on the human side, understanding how to design languages that are easy or productive to use.... The human and computer parts of programming languages have developed in radical asymmetry.”
Observations:

- Intuitions of programmers about what would be helpful are often wrong
- Need to do studies of programmers
Goal: Gentle Slope Systems

Program Complexity and Sophistication

Low Threshold

Difficulty of Use

Goal

Web Development

Java

Visual Basic

Flash

Server-side

C or C# Programming

C Programming

Swing

CSS & HTML

Basic

ActionScript

Small

editor
Making Programming Easier

• Key focus of most of my research
• Life-long pursuit
HANDS

• John Pane, PhD 2002
• **HANDS**
  Human-Centered Advances for Novice Development of Software
Lapis

Rob Miller, PhD 2002

**LAPIS**

Lightweight Architecture for Processing Information Structure

- Informal parsing and reuse of semi-structured text databases
- Simultaneous editing
- Outlier finding
Whyline

- Andy Ko, PhD, 2008
- **WHYLINE** Workspace that Helps You Link Instructions to Numbers and Events

- Causes of software defects
- Understanding software maintenance tasks
- Working sets when programming
- Navigation takes 35% of the time
Crystal

• Ask “why” for everything else!
• **CRYSTAL**
  Clarifications
  Regarding
  Your
  Software using a
  Toolkit,
  Architecture and
  Language
Understanding Unfamiliar Code

- Thomas LaToza, PhD 2012
- Hard to answer questions about code:
  - *Control flow* = “reachability”
  - 100s of others
- “Reacher” visualizes exactly the paths of interest
Backtracking During Editing

- YoungSeok Yoon, PhD 2015
- **AZURITE**
  Adding Zest to Undoing and Restoring Improves Textual Exploration
Observations

- Viewing the code is central!
  - Visualizations of code are mostly useful as navigation aides

- Search is useful across many dimensions, e.g.:
  - Search along control flow
  - Search backwards in time
API Usability

- Jeff Stylos, PhD 2009
- **Application Programming Interface**
  - Libraries, frameworks, SDKs, …
  - Now: web services, middleware, …
  - **User interface** between programmer and functionality in code
- Patterns that decrease usability
- New documentation tools to compensate
<table>
<thead>
<tr>
<th><strong>MICA</strong></th>
<th><strong>JADEITE</strong></th>
<th><strong>CALCITE</strong></th>
<th><strong>EUKLAS</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Makes Interfaces Clear and Accessible</td>
<td>Java API Documentation with Extra Information Tacked-on for Emphasis</td>
<td>Construction And Language Completion Integrated Throughout Eclipse</td>
<td>Eclipse Users’ Keystrokes Lessened by Attaching Samples</td>
</tr>
</tbody>
</table>

**GRAPHITE**

GRAphical Palettes Help Instantiate Types in the Editor

**APATITE**

Associative Perusing of APIs That Identifies Targets Easily

**DACITE**

Design Annotations for Complementing Interfaces Targeting Effectiveness

© 2017 – Brad A. Myers
Glacier and Obsidian

- **GLACIER**
  - Great Languages
  - Allow Class
  - Immutability Easily and Readily

- **OBSIDIAN**
  - Object-oriented Blockchain
  - State Interaction and Development
  - Implementation And Notation

- Michael Coblenz’s PhD work
- Key Goal: Design more usable programming languages

- Glacier:
  - Immutability – declare that instances cannot change after constructed

- Obsidian:
  - Easier blockchain programming
InterState

- Stephen Oney, PhD 2015
Gneiss

• Kerry Chang, PhD 2016

• GNEISS
Gathering
Novel
End-User
Internet
Services using
Spreadsheets

© 2017 – Brad A. Myers
Variolite

- Current PhD work of Mary Beth Kery
  - See our paper, Tuesday, 9:30 - 10:50 in Room 111/113

- **VARIOLITE**
  *(come to the talk to see what it stands for!)*

- Exploratory Programming
Observations

• Always make a video – demos will stop working
• Having a group culture, fun "signature", e.g. for acronyms and gemstones, ribbons
• One good idea makes attending a conference or reading an article worthwhile
Thanks!

- Funding from Xerox PARC, UofT Fellowship, DARPA, NSF, NIH, Microsoft, SAP, Google, Adobe, IBM, Apple, and many others
- My wife and family
- My mentors, colleagues, and especially the hundreds of students
  - Bachelors, Masters, and PhD at CMU
Advisee tree

Thanks to: Fred Hohman (gen. 5), Robert Pienta (gen. 5), and Polo Chau (gen. 4) for animation and ribbon design

See: http://tinyurl.com/myersadviseetree
RUBY: Reminiscing about User interfaces by Brad over the Years

Brad A. Myers

For More information:
• Brad’s home page: bradamyers.com
• Brad’s 66 acronyms: http://www.cs.cmu.edu/~bam/acronyms.html
• Brad’s 172 advisees tree: http://tinyurl.com/myersadviseetree