

Informatics 134

Software User Interfaces
Spring 2021

Mark S. Baldwin

baldwinm@ics.uci.edu

5/20/2021

Agenda

1. Upcoming
2. Graphical Toolkits and Accessibility
3. A4: Speculative Story
4. References

Upcoming

Upcoming

- Finish accessibility and toolkits
- Introduce A4
- Next week: The Future of UI
- Keep working on T3 (DUE EOQ, Final Presentations Start 6/01)

Graphical Toolkits and Accessibility

What is accessibility?

Accessibility

Historical Perspective

Transition from the terminal to a GUI

Ushered in personal computing era
(good)

Ushered in the accessibility era (not so
good)



[Wikipedia, 2021, theverge.com, 2021]

Accessibility

Historical Perspective

Although millions of new people were now **able** to understand and make use of computational systems, millions of people were simultaneously **unable** to use and operate new graphical based systems.



[theverge.com, 2021]

The Graphical User Interface Crisis: Danger and Opportunity

“Our intuition tells that the more an interface is optimized for a person who can see, the less useful that computer will be to people who cannot see.”

——[Boyd et al., 1990]

Accessibility

Historical Perspective

According to [Boyd et al., 1990], the problem manifested in two major categories:

- 1 Perceptual: Screen rendering via pixels requires deciphering of graphical information
- 2 Control: Interaction with visual representations of information and manipulation and control of the flow of information



[theverge.com, 2021]

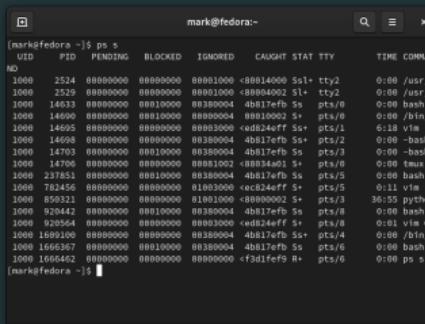
Accessibility

Historical Perspective

WIMP (?):

Transition from a concise command language

To visual metaphors – graphical representations of everyday objects



```
mark@fedora:~]$ ps s
  PID  PPID  STAT   TIME COMMAND
 1000    1    S+  0:00 /usr/r
 1000  2529    S+  0:00 /usr/r
 1000 14633    S+  0:00 bash
 1000 14690    S+  0:00 /bin/
 1000 14693    S+  0:00 pts/0
 1000 14698    S+  0:00 bash
 1000 14703    S+  0:00 bash
 1000 14850    S+  0:00 bash
 1000 23785    S+  0:00 bash
 1000 782456   S+  0:11 vim
 1000 859321   S+  36:55 python
 1000 920442   S+  0:00 bash
 1000 920564   S+  0:01 vim C
 1000 1109106   S+  0:00 bash
 1000 166462   S+  0:00 bash
 1000 166462   S+  0:00 ps s
```

[theverge.com, 2021]

Accessibility

Historical Perspective

Gaining Access: Enter the Screen Reader

IBM Screen Reader/DOS (1984)

IBM Screen Reader/2 (1986-1994)

...

IBM adapted early work on speech synthesis to create SAID, the Synthetic Audio Interface Driver [Thatcher, 1994].

Historical Perspective

Gaining Access: Enter the Screen Reader

Early version of SR/2 relied on a separate custom keyboard to control speech synthesis to avoid system conflicts!

The key principle from Thatchers work was that text-based DOS and GUI are different interfaces for doing the same thing. So, the solution was to map GUI to textual equivalents [Thatcher, 1994].

“Abstract away what is *graphical* about the graphical user interface.”

Accessibility

Historical Perspective

From *access* to *degrees of access*

By mid-1990's focus changed to efficiency, coherence, exploration, and cost

Refinement of the abstraction – developing a consistent, reusable, lexical understanding of graphical widgets

Locate widgets without a mouse, support exploration, interact without clicking

Consistent mental model (WHY?)

Accessibility

Historical Perspective

From *access* to *degrees of access*

The Mercator Project (Georgia Tech, Center for Rehab Technology)

GUIB: Textual and Graphical User Interfaces for Blind People (European Commission, Technology Initiative for the Disabled and Elderly Persons or TIDE)

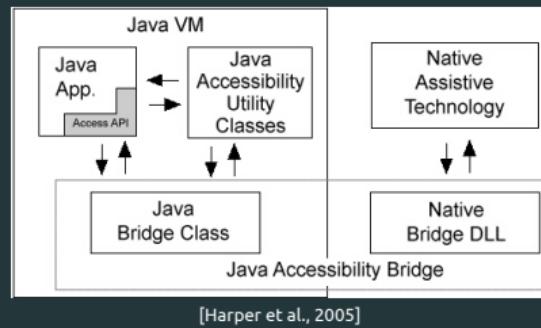
interaction object	example	braille-based presentation	example	speech-based presentation	example	nonspeech-based presentation	example
caret, mouse pointer		one braille character	t■xt ■	text around caret is spoken	"e"	audio "click" at caret	t e click x t
text, text attributes		braille, attributes through dots 7 and 8 or on request	<u>sample</u>	text is spoken, attributes are verbalized (??)	"sample"	pitch of speech is modified for attributes	
window		window frame in braille name is spoken pop-up, move, size by sound	+[-] Notepad - Untitled P File E Edit S Search	name is spoken	"notepad"	auditory icon for window object, modified for popup, iconify, or focus	tapping on glass sound
icon		name in braille	[Icon] Dustbin		"dustbin"	auditory icons	sound of dropping something in a trashcan
menu		all items in braille vertical or horizontal layout	P File O Options W Wi *A Auto Arrange M Minimize on *S Save Setting	new selection is spoken	"auto arrange"	auditory icon for menu-button, pitch is modified relative to location in menu	flipping sound at a high pitch
scroll bar		in braille	■□-----■	status is verbalized	"slider at zero percent"	auditory icon for scrollbar, location conveyed with pitch	slide whistle sound, low pitch
edit field		in braille, selection with dots 7 and 8	winword.exe	text is spoken	"winword dot exe"	auditory icon for editable text field	sound of an old-fashioned typewriter
list box		all items in braille	c:\batch bc dict dict dos dustbin	selection is spoken	"c colon backslash"	auditory icon for list, pitch is modified relative to location in list	line-printer sound at a high pitch
button		in braille	[Y Yes]		"yes"	auditory icon for push button	sound of pushing an old elevator button

Accessibility

Historical Perspective

From *access* to *degrees of access*

Projects like Mercator and GUIB influenced the design and implementation of accessibility API's in programming languages like Java, .Net, Cocoa (OSX)

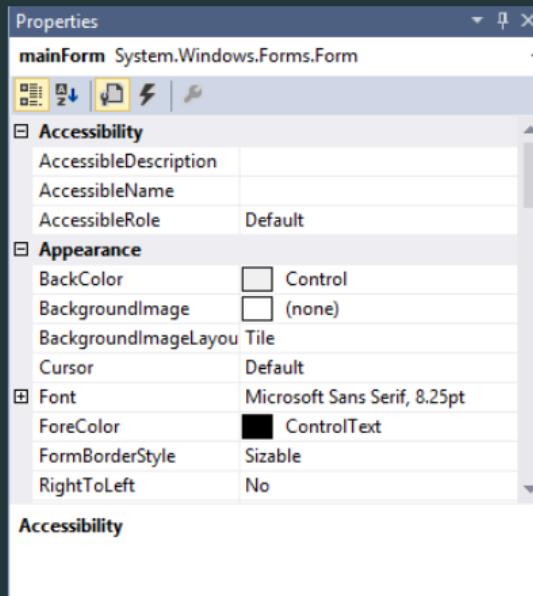


Accessibility

Historical Perspective

From *access* to *degrees of access*

An example from Microsoft Visual Studio



When converted to code:

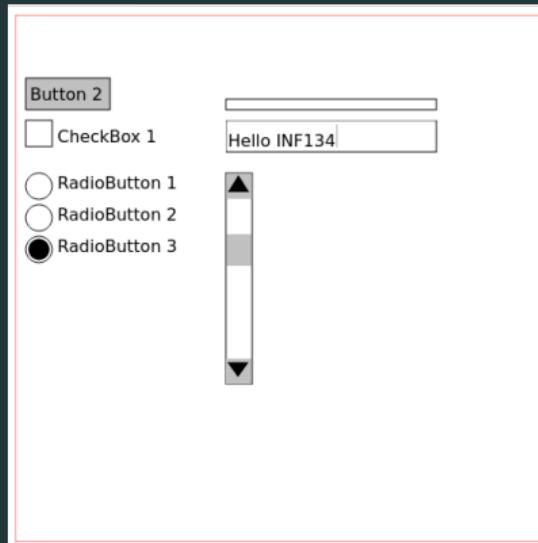
```
1 //describes control  
2 mainForm.AccessibleDescription = "the main form";  
3 //name reported to accessibility aids  
4 mainForm.AccessibleName = "My Program";
```

[learn more](#)

Accessibility

Historical Perspective

How would you make your widgets accessible?



From *degrees of access* to *supporting access*

Accessibility

Supporting Access

The WebAIM Million: 2019-2021

Over 51 million a11y errors, avg of 51.4 errors per page

97.4% of home pages had detectable WCAG2 failures

65% missing alt text for images

25% empty buttons or button misuse

**Today we *have* the ability to make software accessible,
but we lack the *desire or awareness* to make it so.**

Why is the second example *more* accessible?

```
1 <div>Next</div>
```

```
1 <button>Next</button>
```

Why is the second example *more* accessible? Semantics

```
1 <div>Next</div>
```

```
1 <button>Next</button>
```

Which example is *harder* to implement?

Accessibility

Semantic HTML

They require the same amount of effort :) Plus:

Easier to reason about, less code (don't have to add accessibility)

Therefore...lighter/faster/better optimization (SEO, etc.)

Browser accessibility engine translates accessible functions automatically
(e.g., Tab and Enter keys, purpose)

see: [HTML: A good basis for accessibility](#) for a more detailed overview.

The Visuospatial Sketchpad

Making graphical interfaces accessible is more than just semantics. In his work on human working memory, Alan Baddeley [Baddeley, 1992] identifies the visuospatial sketchpad as:

“virtual environment for physical simulation, calculation, visualization, and optical memory recall.”

Move optically retained information from short-term to long-term memory and back again

Think about how humans recognize faces

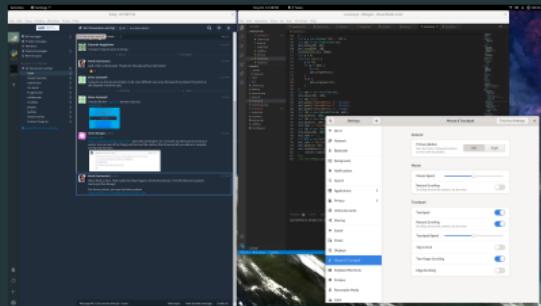
visual cache - holds information about form and color

inner scribe - manages spatial and movement information

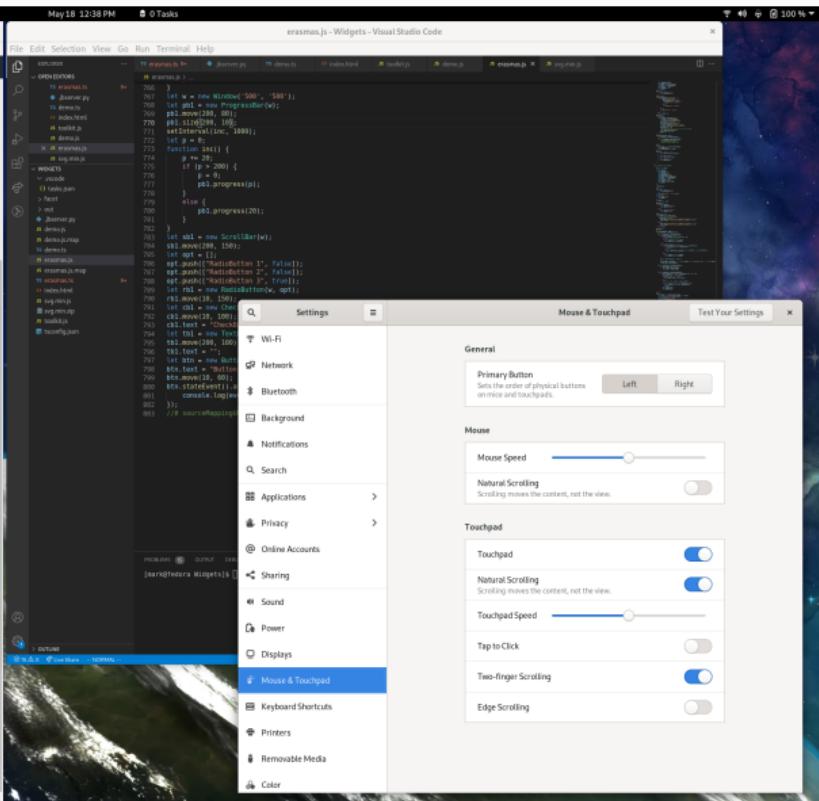
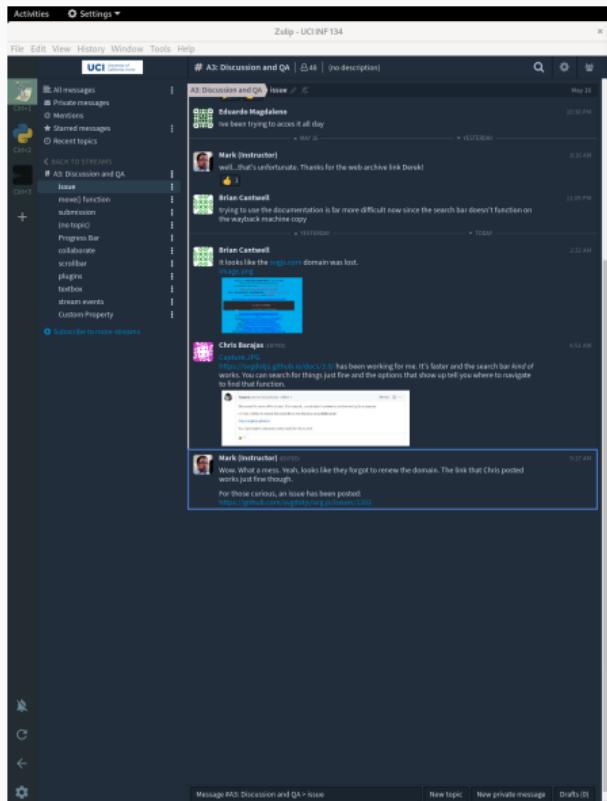
Accessibility

The Visuospatial Sketchpad

Take a minute to study the screen shot pictured here. Can you identify some ways that information is communicated visually?



Remind Mark: Full screen image on next slide ;)



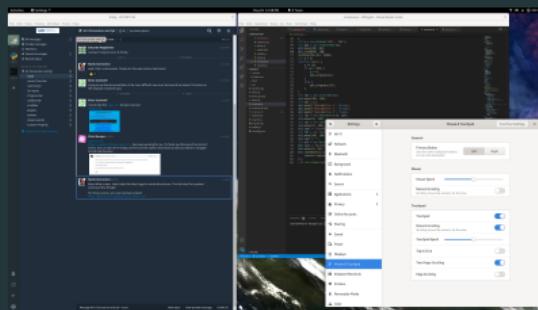
Accessibility

The Visuospatial Sketchpad

Visual glance

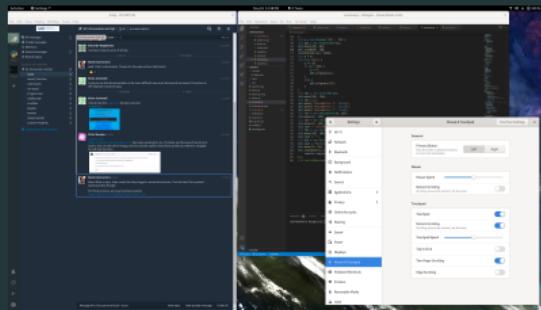
Spatial arrangement

Color



Accessibility

How do we make these visual enhancements more accessible?



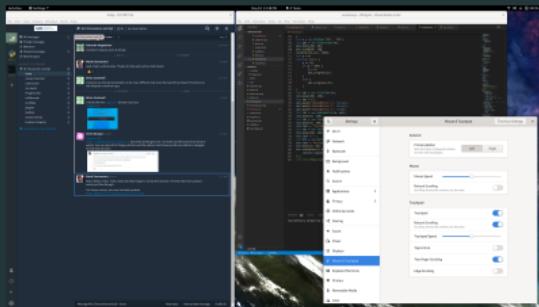
Accessibility

How do we make these visual enhancements more accessible?

Hierarchy!

Shortcuts (skip links, etc.)

Color alternatives, high contrast



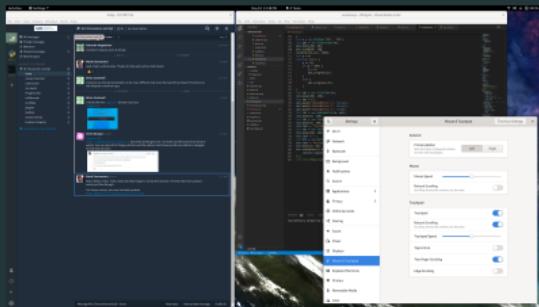
Accessibility

How do we make these visual enhancements more accessible?

Hierarchy!

Shortcuts (skip links, etc.)

Color alternatives, high contrast



Accessibility

How do we make these visual enhancements more accessible?

Some thoughts from my own work...

Semantic versus lexical

Semantic information as haptic information

Visuospatial to tangible



Demo Video

Accessibility

Let's return to the topic of supporting access...

**Today we *have* the ability to make software accessible,
but we lack the *desire or awareness* to make it so.**

A4: Speculative Story

A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces...

A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

Wearable Technologies



A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

Wearable Technologies

Augmented Reality



A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

Wearable Technologies

Augmented Reality

Virtual Reality



A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

Wearable Technologies

Augmented Reality

Virtual Reality

Voice Assistants



A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

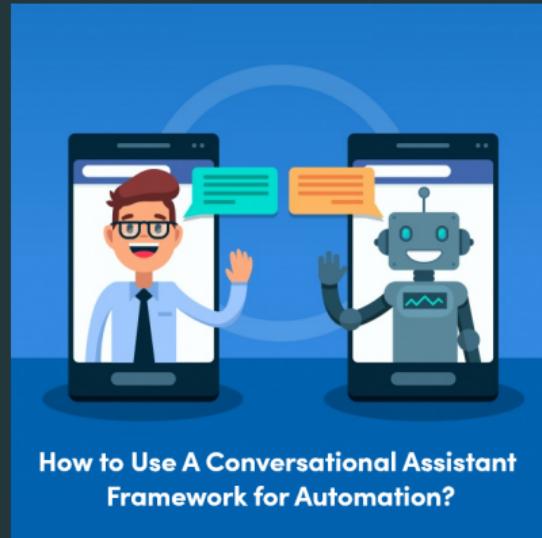
Wearable Technologies

Augmented Reality

Virtual Reality

Voice Assistants

Conversational Agents



A4: Speculative Story

Envision a toolkit for tomorrow's user interfaces

Wearable Technologies

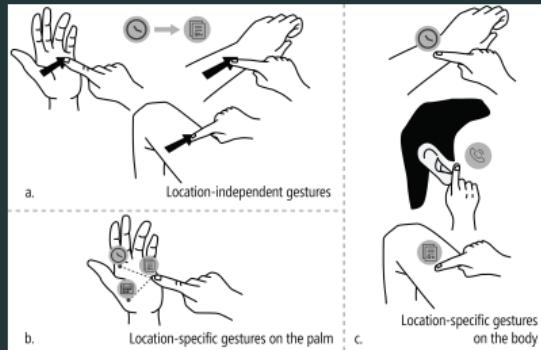
Augmented Reality

Virtual Reality

Voice Assistants

Conversational Agents

Future advances (eyewear, on-body interaction, projected displays, etc.)



A4: Speculative Story

What will the toolkits, widgets, and interactions look like for these types of systems?

A4: Speculative Story

What will the toolkits, widgets, and interactions look like for these types of systems?

(show them an example, Mark)

Pick a speculative interface and envision *three* novel interactors to control your interface

No programming!

Deliverable: Slide Deck (5-10 slides)

Develop assets to document and demonstrate your interactors

References

References i

-  Baddeley, A. (1992).
Working memory.
Science, 255(5044):556–559.
-  Boyd, L. H., Boyd, W. L., and Vanderheiden, G. C. (1990).
The graphical user interface: Crisis, danger, and opportunity.
Journal of Visual Impairment & Blindness, 84(10):496–502.
-  Harper, S., Khan, G., and Stevens, R. (2005).
Design checks for java accessibility.
-  Mynatt, E. D. and Weber, G. (1994).
Nonvisual presentation of graphical user interfaces: Contrasting two approaches.
In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, page 166–172, New York, NY, USA. Association for Computing Machinery.

References ii

-  Thatcher, J. (1994).
Screen reader/2: Access to os/2 and the graphical user interface.
In *Proceedings of the First Annual ACM Conference on Assistive Technologies, Assets '94*, page 39–46, New York, NY, USA. Association for Computing Machinery.
-  theverge.com (2021).
40 years of icons: the evolution of the modern computer interface.
-  Wikipedia (2021).
Computer terminal.