

Informatics 134

Software User Interfaces
Spring 2023

Mark S. Baldwin

baldwinm@ics.uci.edu

05/01/2023

Agenda

1. Upcoming
2. Mid-Quarter Presentations
3. Graphical Toolkits and Accessibility
4. References

Upcoming

Upcoming

Agenda

Today:

- Discussion on complex widget design
- Accessibility and Toolkits lecture

Wednesday:

- Mid-quarter Presentations!
- Refresher Quiz 1 (weeks 1-4)

Next Week:

- Keep working on A2 (DUE 5/8)
- Keep working on team project

Mid-Quarter Presentations

Mid-Quarter Presentations

Presentation Format

Submit a link to a Google Slides deck. Your deck must include 1 slide that includes the following:

Your team name and your names

Some sort of visualization:

- A sketch of your product UI

- A storyboard or other asset you created

- A 'mood' board of similar UI's that convey what you are trying to build

A few words or bullet points that cover the following:

- What you've done so far

- What you plan on doing

Challenges and obstacles so far

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]

Accessibility

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]

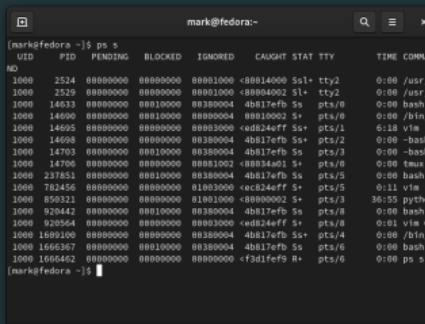
Accessibility

Historical Perspective

WIMP (?):

Transition from a concise command language

To visual metaphors – graphical representations of everyday objects



A screenshot of a terminal window titled "mark@fedora:~". The window displays the output of the command "ps -s". The table has columns: NO, PID, PENDING, BLOCKED, IGNORED, CAUGHT, STAT, TTY, TIME, and COMA. The data shows various processes running, such as bash, vim, and python, along with their respective process IDs and states.

NO	PID	PENDING	BLOCKED	IGNORED	CAUGHT	STAT	TTY	TIME	COMA
1000	3534	00000000	00000000	00001000	<80014000	Ss+ tty2		0:00	/usr/
1000	2529	00000000	00000000	00001000	<80004002	S+ tty2		0:00	/usr/
1000	14633	00000000	00010000	00380004	4b817efb	Ss pts/0		0:00	bash
1000	14690	00000000	00010000	00380004	88100002	S+ pts/0		0:00	/bin/
1000	14693	00000000	00000000	00003004	<d824ef	F S+ pts/1		6:18	vim
1000	14698	00000000	00000000	00380004	4b817ef	S+ pts/2		0:00	bash
1000	14703	00000000	00010000	00380004	4b817efb	Ss pts/3		0:00	bash
1000	14708	00000000	00010000	00380004	4b817efb	Ss pts/4		0:00	bash
1000	237853	00000000	00010000	00380004	4b817efb	Ss pts/5		0:00	bash
1000	782456	00000000	00000000	01083004	<c824ef	F S+ pts/5		0:11	vim
1000	859321	00000000	00000000	01081000	<80000002	S+ pts/3		36:55	python
1000	920442	00000000	00010000	00380004	4b817efb	Ss pts/8		0:00	bash
1000	920564	00000000	00000000	00003000	<d824ef	S+ pts/8		0:01	vim C
1000	1109106	00000000	00000000	00380004	4b817efb	S+ pts/4		0:00	/bin/
1000	10000007	00000000	00010000	00380004	4b817efb	Ss pts/6		0:00	bash
1000	1664462	00000000	00000000	00000000	<fd10ef9	H pts/0		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)

...

James Thatcher at 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)

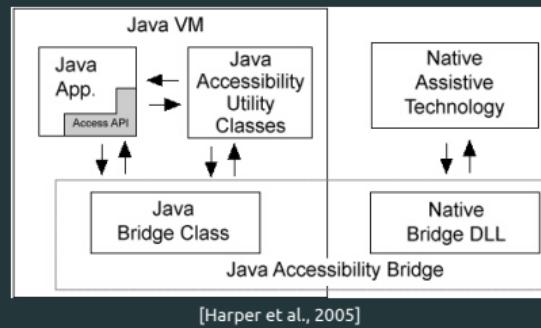
interaction object	example	braille-based presentation	example	speech-based presentation	example	nonspeech-based presentation	example
caret, mouse pointer		one braille character	t■xt ■	text around caret is spoken	"e"	audio "click" at caret	t e click x t
text, text attributes		braille, attributes through 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)

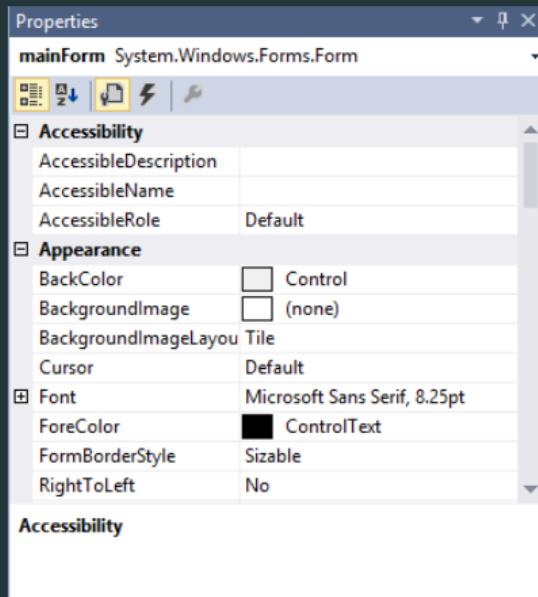


Accessibility

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](#)

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

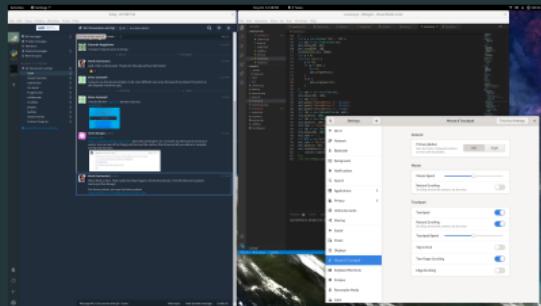
visual cache - holds information about form and color

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 ;)

The image shows a desktop interface with several windows open:

- Activities**: A sidebar with navigation links like All messages, Private messages, Starred messages, Recent topics, Back to streams, and a plus sign for adding more elements.
- Zulip - UCI INF134**: A messaging application window titled "# A3: Discussion and QA". It shows a conversation between users Edwardo Magdaleno, Mark (Instructor), Brian Cantwell, and Chris Borja. The messages discuss domain access and a screenshot of a browser showing a broken link.
- May 18 12:38 PM**: A terminal window titled "erasmus.js - Widgets - Visual Studio Code". It displays a portion of a JavaScript file with code related to window positioning and scroll bars.
- Mouse & Touchpad Settings**: An open dialog box for adjusting mouse and touchpad preferences. It includes sections for General (Primary Button, Left, Right), Mouse (Mouse Speed, Natural Scrolling), and Touchpad (Touchpad, Natural Scrolling, Touchpad Speed, Tap to Click, Two-finger Scrolling, Edge Scrolling).

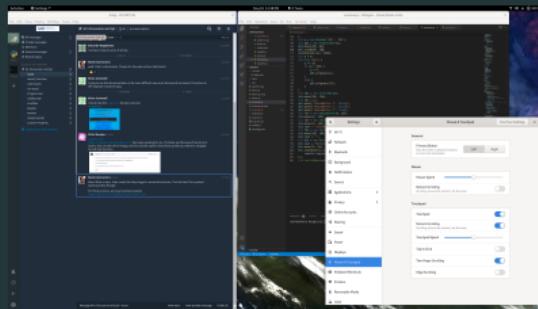
Accessibility

The Visuospatial Sketchpad

Visual glance

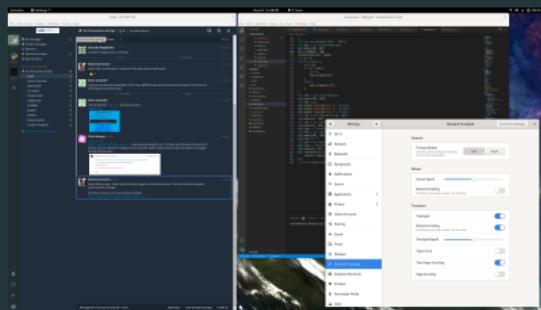
Spatial arrangement

Color



Accessibility

How do we make these visual enhancements more accessible?



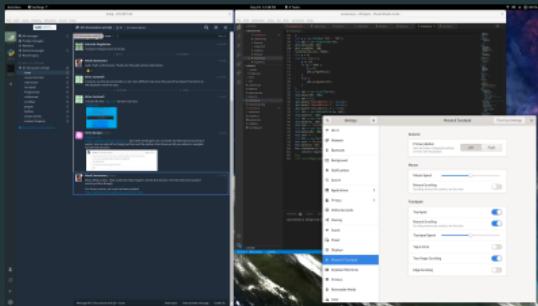
Accessibility

How do we make these visual enhancements more accessible?

Hierarchy!

Shortcuts (skip links, etc.)

Color alternatives, high contrast



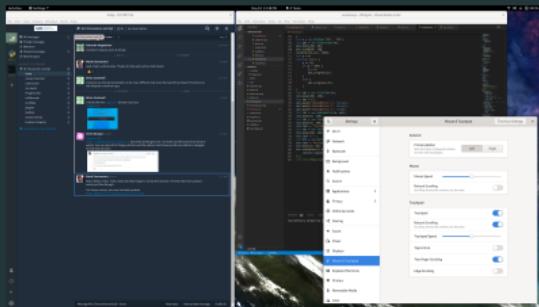
Accessibility

How do we make these visual enhancements more accessible?

Hierarchy!

Shortcuts (skip links, etc.)

Color alternatives, high contrast



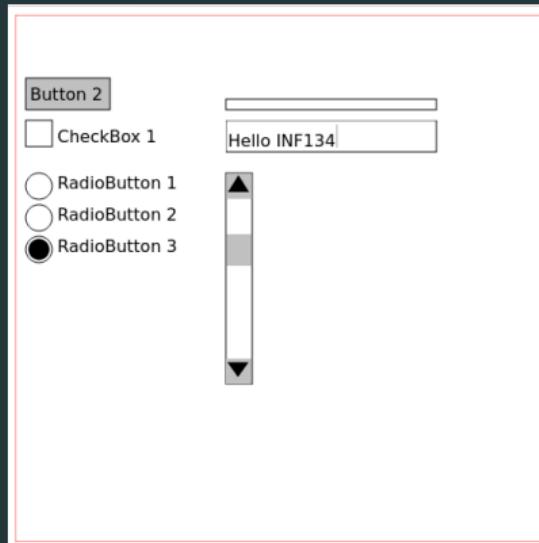
Demo Video

<https://vimeo.com/231640164>

Accessibility

Supporting Access

How can we make our custom toolkit widgets accessible?



Supporting Access

Based on SVG, so at a minimum follow web standards

Rely on Aria roles with custom widgets

```
1 enum RoleType {  
2     button = "button",  
3     checkbox = "checkbox",  
4     ...  
5 }  
6  
7 interface IAccessibility {  
8     set role(role: RoleType);  
9     get role(): RoleType;  
10 }
```

Supporting Access

Based on SVG, so at a minimum follow web standards

Rely on Aria roles with custom widgets

```
1 // set Aria role
2 class Button extends Widget{
3     constructor(parent:Window){
4         this.role = RoleType.button;
5     }
6 }
7
```

```
1 <svg>
2     <g role="button" tabIndex="2">
3         <rect ... ></rect>
4         <text>
5             <tspan ...>Click Me</tspan>
6         </text>
7     </g>
8 </svg>
```

Team Activity

How does your chosen toolkit (either one) support accessibility?

Accessibility

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

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

Access to technology is democratizing. It is our responsibility, as creators of technology to accommodate all people, regardless of ability.

Accessibility

What are some ways that you can be an ally and advocate for accessible design?

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.