



# Access by Contract

## The Third Generation Approach to Supporting People with Disabilities

Peter Korn

Accessibility Architect

Sun Microsystems, Inc.



# Why are we here?

# 1<sup>st</sup> Generation Accessibility: late 1960s, 1970s, early 1980s

- Blind access with speech, Optacon
- Low vision access via special hardware
- Early Braille printers
- Talking calculators



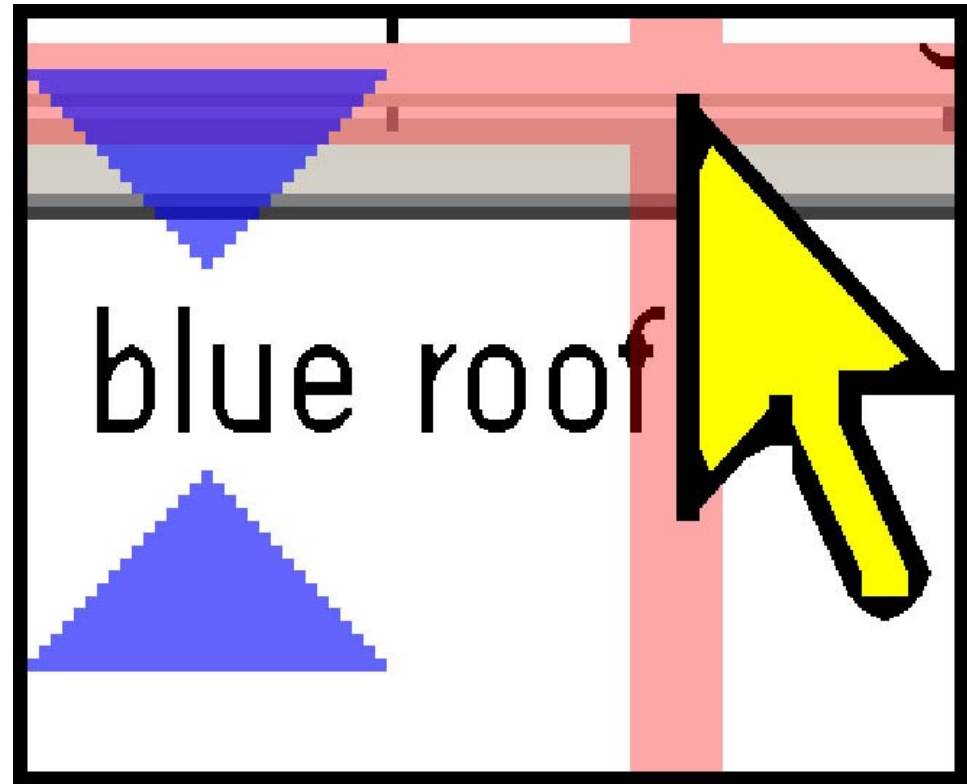
# 1<sup>st</sup> Generation Limitations

- Options expensive, difficult to use (such as Optacon)
- Computing limits restricted what users could do – only job access; no Web, no on-line books
- Very few disabilities served
  - No augmentative communication devices
  - Poor options for physical impairments
  - No voice recognition
  - Nothing for cognitive impairments

# 2<sup>nd</sup> Generation Accessibility:

late 1980s, 1990s, early 2000s

- Software TTS, access to the GUI, scripting
- Software mag
- Voice recognition, OCR, Aug Comm
- WYNN, TextHelp, other LD products



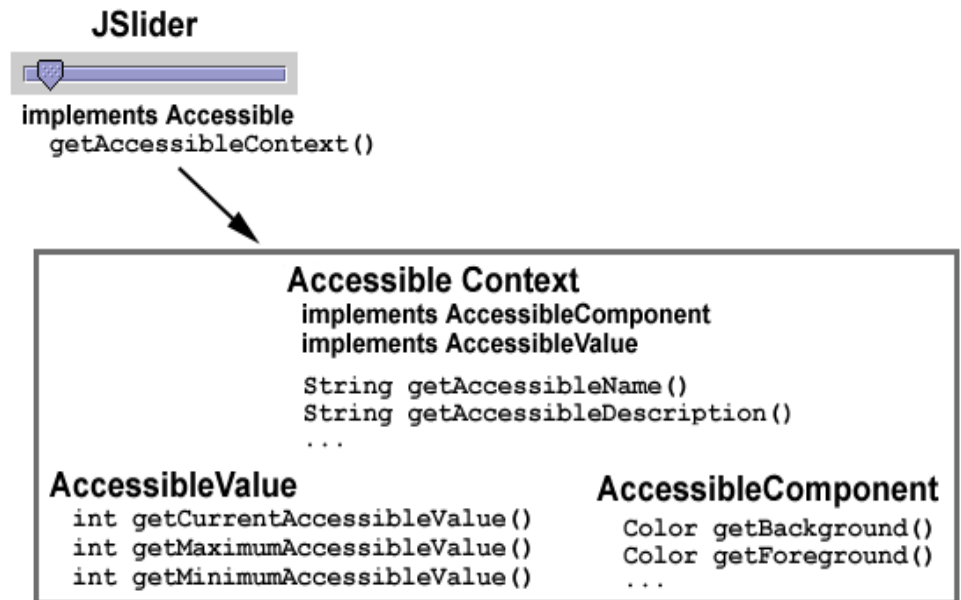
## 2<sup>nd</sup> Generation Limitations

- Patching of OS display drivers, keyboard driver means solutions are brittle: must update every OS release
- Special case work for applications – results in brittle solution: must update ever app update
- Many apps bypass patching mechanisms, so are inaccessible or need lots of special work:
  - HyperCard, DirectX, X Servers, Terminal Server

**Note: Section 508 developed in this environment**

# 3<sup>rd</sup> Generation Accessibility: 1997 and onward

- Access by Contract
- Every UI element implements it
- Everything needed by all AT provided
- Rich, extensible, flexible, powerful



# 3<sup>rd</sup> Generation Accessibility: How it came to be; why needed

- HyperCard, Macintosh X Servers → Access Aware
- Web Accessibility → WCAG, UAAG, ATAG
- Microsoft Office → MSAA
- Java via Direct X & own text rendering → JA-API
- UNIX & X → GNOME Accessibility API

# It is where we are all going

- Microsoft Windows
  - UI Automation in Longhorn: stock UI, MS apps support
- Apple Macintosh
  - Accessibility APIs: supported by Carbon, Cocoa
  - Mac OS X v.10.4 to include AT that uses these
- Java platform
  - Java Accessibility API: in Swing, Oracle UI toolkits, others
  - Already used by JAWS, ZoomText, Virgo
- UNIX (Sun Solaris, GNU/Linux, others)
  - GTK+, OpenOffice, Mozilla, Qt, Java all support accessibility
  - Already used by Gnopernicus, GOK, Dasher

# 3<sup>rd</sup> Generation Implications

- Platform has a formal responsibility for access
  - Define a comprehensive, extensible Accessibility API
  - Implement that API in all UI toolkits
- Applications have a clear interface to support
  - No more guessing, finger pointing
  - No more “works with JAWS” as “good enough”
- AT no longer responsible for everything
  - No more patching, reverse engineering
  - No need to update every time the OS or apps change

# 3<sup>rd</sup> Generation Benefits...

- ... for the mainstream software industry:
  - Clarity, testability, formality
- ... for the AT vendor:
  - Can focus on the user, not on reverse engineering
  - Lower barrier of entry in developing new AT; F/OSS AT
- ... for the user with a disability:
  - More features in AT; more kinds of AT (new LD apps!)
- ... for U.S. Government procurement:
  - Move toward testability of mainstream software
  - Emergence of standards (see <http://www.a11y.org>)

# What this means for Sec 508

- Brilliance of “Equivalent Facilitation” means we can move to the 3<sup>rd</sup> Generation world immediately
- Is it time to review some technical standards?
  - 1194.21(d) “Sufficient information about a user interface element including the identity, operation and state of the element shall be available to assistive technology.”
  - 1194.21(f) “Textual information shall be provided through operating system functions for displaying text.”
- Suggests a larger review of 508 Subpart B
  - Specific section for an app platform, for a UI toolkit

# Concluding Thoughts...

- The shift from 1<sup>st</sup> generation to 2<sup>nd</sup> generation wasn't easy or comfortable
- The shift from 2<sup>nd</sup> generation to 3<sup>rd</sup> generation probably won't be smooth either
- Benefits of 3<sup>rd</sup> generation are enormous
- We can demand better: from our OS vendors, from our mainstream applications, and from our Assistive Technologies



Peter Korn

[peter.korn@sun.com](mailto:peter.korn@sun.com)

