Accessibility in the Cloud

*Boldly venture forth, ye brave explorers!*

**Jonas Klink** (klink@google.com)
Accessibility Product Manager / Software Engineer

http://google.com/Accessibility
Who am I?

- PM/SWE, part of a Dev team dedicated to Access Engineering
- With Google for the past 4 years
- Education
  - MS in Computer Science (Chalmers Uni. of Tech., Sweden)
  - PhD in Computer Science (University of Washington, Seattle)
- Research on Education and Technology for the visually impaired
- Project work includes:
  - Client-side: Toolbar, Desktop Search, Chrome
  - Web Apps: Gmail, Apps, Blogger, Maps, Transit, …

http://google.com/Accessibility
Why? Users!

Internet Population

- English
- Chinese
- Color deficient
- Japanese
- Spanish
- Poor vision
- Poor dexterity
- German
- French
- Korean
- Deaf
- Italian
- Portuguese
- Russian
- Arabic
- Dutch
- Swedish
- Blind

Millions of users

http://google.com/Accessibility
The Web as a Platform

http://google.com/Accessibility
Please ask staff for help

http://google.com/Accessibility

FAIL!
The Web as a Platform

- Platform layers are changing
  1. Low-level support framework (TTS, fonts, themes)
  2. JavaScript APIs
  3. Web Applications (GWS, Gmail, Docs)

- Graceful Degradation vs. Progressive Enhancement

- The Web has the distributed data
  - Universal Access Engineering makes it available through any channel

- Personalization and user goals are key
  - Every level in the stack is customizable
  - APIs provides the muscle
  - User is less dependent on the applications

http://google.com/Accessibility
Designing for Access Workflows

- Focus on workflows, rather than UI components
  - Most common tasks need to be optimized
  - Tab/arrow navigation often too slow
  - Enumerating workflows often highlight common roadblocks
  - Workflows drill down to component-level access

- Designing your product for optimized workflows
  1. Optimize workflows with keyboard and AT support
  2. Expose a public page-level API, addressable from JS
  3. Provide a clean DOM, with non-obfuscated hooks

- Document and empower the curious user!

http://google.com/Accessibility
Supporting the brave explorers

○ Exploration in the document model Web
  ■ Web 1.0: Headings, links, frames to build cognitive model
  ■ AT optimized for quick access to key element types
  ■ Users have developed personalized techniques for exploration

○ Need support for exploration in Web 2.0
  ■ Web 2.0: application mode and non-document structure
  ■ Sighted users rely on visual cues learned from the desktop
  ■ Answer: contextual, on-demand exploration aids?
  ■ Community can work together to build familiarity

http://google.com/Accessibility
Example: Google Reader Access

- **Extremely keyboard friendly**
  - Access keyboard shortcut through '?' or Reader Help Center
  - Navigate items with 'j' and 'k'
  - Keyboard bindings available for starring, sharing, commenting, etc

- **Delivers screen reader augmentation**
  - Follow link 'click here for ARIA enhanced Google Reader'
  - Screen reader support in ARIA-enabled browsers

- **Applies magnification lens for low-vision users**
  - Follows keyboard navigation
  - Provides customization through '-' and '='

- **Zero impact on latency!**

http://google.com/Accessibility
Conclusion

Collaboration and openness benefit everyone

Customization is key
Configure once, work everywhere

Focus on workflows, then widgets

Develop solutions with little or no latency impact
Thank you!

Q & A

google.com/accessibility

klink@google.com

http://google.com/Accessibility