Enhancing Document Accessibility

Response to Call for Bids

GNOME Foundation

A Coruña, 9th March 2013

Contents

1 Project Overview

This project consists of improving the accessibility support of documents in GNOME, with an emphasis on making it possible for users who are blind or visually impaired to read and interact with PDFs and other document content using GNOME tools.

1.1 Goal

Upon completion of this work, Orca users will be able to independently read the content of, and fill out forms in, accessible[1] PDF documents. Focus will be placed on providing full, compelling access to tagged PDFs[2] as they offer the most accessibility-related information. The accessibility of non-tagged PDFs will be improved to the greatest extent possible.

The GNOME tools for accessing PDF documents will include Evince, Epiphany/Web, and GNOME Documents. The U.S. Government's IRS W-4 form[3] will be used as the primary benchmark by which to judge the accessibility improvements to PDF documents in GNOME. Additional untagged PDF documents will be located and/or created so that the improvements made during this project towards these documents can also be assessed.

1.2 Objectives

In order to achieve the goal stated above, a number of objectives must be met:

- PDF documents and the tools which embed them must:
  - Be fully keyboard navigable, both natively and through assistive technologies
  - Emit the expected accessibility events/signals which make it possible for assistive technologies to respond appropriately as a user navigates
- PDF document content must be exposed to assistive technologies:
  - In its entirety, in discrete objects, and in the correct order
  - With appropriate attributes and interfaces
  - With as much structural and semantic information possible

A list of specific tasks to be performed for each objective, as well as a description of the current status, is provided in the sections which follow.

[1] "Accessible" here should be taken to mean documents which consist of actual text which the renderer can obtain. PDFs consisting of scanned images of text require conversion via optical character recognition and are thus beyond the scope of this project.

[2] A "tagged" PDF document is one that includes structural and semantic information which, if exposed to assistive technologies, greatly improves the accessibility of the content. While tagged PDFs are not widespread, they are often found in documents which are required by law to be accessible.

[3] `http://www.irs.gov/pub/irs-pdf/fw4.pdf`

2 Current Status

2.1 History and Summary of Findings

Basic AtkText functionality was added to Evince in 2010 as part of Lot 3 of the Guadalinfo Accessible Project[1]. However, Evince documents continued to be completely inaccessible until January 2013[2] when EvViewAccessible's use of AtkObjectFactory was removed[3]. This change resulted in the document object and its AtkText implementation being available to assistive technologies. Shortly thereafter, support for AtkHypertext and AtkHyperlink was added to Evince[4].

Unfortunately, the work completed thusfar is insufficient for Orca to provide even a minimal level of support for reading simple PDF documents in Evince. When GNOME 3.8 is released:

- The textual content of PDF documents viewed in Evince will barely be accessible.
- Non textual content, including forms, will be completely inaccessible.
- Embedded PDFs, including LibreOffice files viewed in GNOME Documents, will inherit the above accessibility issues.

The other tools of GNOME Documents will be fairly accessible, but a number of improvements are recommended for more compelling access in subsequent GNOME releases.

2.2 Evince

2.2.1 Reading Text

Findings: Access to text is minimal and limited to single-column documents.

Barring the use of conversion tools such as pdf2html and OCRFeeder, the only PDF content which will be accessible to users who are blind is the text from single-column documents. And the only means to access that small subset of content is to fall back on Orca's Flat Review feature[5] whose "bird's eye view" presentation was never intended for, and is ill-suited to, document access. As a result, text which is scrolled off-screen will require the user who is blind to first scroll the desired text on screen before it can be read; text contained in multiple columns will be presented as if it was contained in a single column rendering it unintelligible.

Tabular data, while not entirely inaccessible, may nonetheless be quite difficult to comprehend non-visually due to the absence of the table's structure and layout, which are normally exposed through the AtkTable interface, along with the possibility that its contents have been rendered in the wrong order.

[1] https://live.gnome.org/Guadalinfo_accesible
[2] https://bugzilla.gnome.org/show_bug.cgi?id=677348
[3] https://bugzilla.gnome.org/show_bug.cgi?id=685828
[4] https://bugzilla.gnome.org/show_bug.cgi?id=639403
[5] https://help.gnome.org/users/orca/stable/howto_flat_review.html

Text selection is inaccessible: Lack of caret navigation in Evince prevents users who are blind from selecting portions of a document. In addition, Evince's AtkText implementation, which includes selection support, fails to emit the expected signals and fails to report the selected text when queried. There is a patch proposed to solve the caret-navigation issue.[6]

Text attributes are also inaccessible. While this issue is not nearly as significant as the others found during our assessment, it does prevent users who are blind from being able to confirm that the documents they create have the appearance they intend. There is a patch proposed to solve this issue.[7]

2.2.2 Using Screen Reader Enhanced Navigation

Findings: Screen reader enhanced navigation is not possible.

For users who are blind, it is often desirable to also be able to navigate amongst document elements such as headings, paragraphs, and tables, in much the same way that sighted users visually scan through headings to locate paragraphs about a desired topic. For these reasons, screen readers typically[8,9,10] include navigational commands to move more quickly amongst these objects than native caret navigation provides (e.g. jump to the next heading). Orca has these features in place,[11] but because the entire document is exposed as a single accessible text object and because the caret cannot be (re)positioned via AtspiText, navigation amongst elements is not possible.

2.2.3 Accessing Embedded Objects

Findings: Objects embedded within the text are not accessible.

Embedded objects such as images will go undetected by the user because these objects are neither present in the accessible hierarchy nor represented by "replacement characters" in the accessible text. While links are detectable now that AtkHypertext and AtkHyperlink have been implemented in Evince, the fact that links are neither keyboard focusable nor discrete accessible objects means links in PDFs will likely go unnoticed by Orca users.

2.2.4 Filling Out Forms

Findings: Filling out forms is not accessible.

Users cannot move to form fields to fill them out without using a mouse. Orca cannot compensate for this even with its Flat Review feature because the widgets are not present amongst the accessible objects. When one interacts with these objects, the objects fail to emit the accessibility signals needed for Orca to provide confirmation to the user who has given a form field focus, toggled its state, changed its contents, moved the caret, selected text, etc.

[6] https://bugzilla.gnome.org/show_bug.cgi?id=638905
[7] https://bugzilla.gnome.org/show_bug.cgi?id=639932
[8] http://www.apple.com/accessibility/voiceover/
[9] http://www.freedomscientific.com/doccenter/archives/training/JAWSKeystrokes.htm
[10] http://www.nvda-project.org/documentation/userGuide.html#toc37
[11] http://help.gnome.org/users/orca/stable/commands_structural_navigation.html

2.3 GNOME Documents

2.3.1 Locating Documents

Findings: The list of available documents and search functionality is largely accessible, but there are some areas in which improvements are recommended.

Both the list and the grid views in which documents are displayed are keyboard navigable, and Orca presents each item navigated to in response to object:state-changed:focused events. Orca does not, however, present the displayed author because this information is not exposed to assistive technologies. In addition, because each item has ATSPI_STATE_SELECTABLE even though it cannot physically be selected/highlighted, Orca speaks an unwanted "unselected" as the user navigates from item to item.

The document search functionality is largely accessible, but some additional work will be needed in both Orca and in GNOME Documents for this feature to be compellingly accessible to Orca users. In particular:

- When the search entry is given focus, Orca only presents it as "text" because this entry lacks an accessible name.
- When the results update, Orca presents nothing because focus remains in the search entry and the accessible events received are insufficient for a generic/Orca-wide heuristic identification that search results should be presented.
- Orca does not present the search options table contents very well. This is not a result of this widget being inaccessible, but instead is simply a new use case for accessible tables (namely functioning like a menu) which Orca does not yet handle.
- There are some occasional problems with keyboard navigation, such as not always being able to give focus back to the documents list via Tab. But this issue is not reliably reproducible and more investigation is needed.

2.3.2 Viewing Documents

Findings: Access to content depends on Evince accessibility. Accessibility of navigation within the viewer needs to be improved.

All documents which can be viewed in GNOME Documents are displayed via an embedded Evince document. As a result, the current level of document accessibility found in Evince is the same level of document accessibility found in GNOME Documents' document view, and improvements made to Evince should automatically carry over to GNOME Documents.

The document's view navigation bar widgets are presented by Orca as the user moves amongst them, though this presentation would be improved by the addition of accessible names to those widgets. More notably, however, is the fact that there does not seem to be any way to make the navigation bar appear other than clicking in the document area with the mouse.

Changing pages with the keyboard without using the navigation bar is possible via Page Up/Page Down, but only after the user has clicked somewhere on the page.

There are two occasions when Orca fails to present anything within the document view: When pages change and when returning to the list of documents. In both cases the problem is caused by the lack of the accessible events expected by Orca. These issues can be resolved either within Orca or within GNOME Documents and the available solutions should be discussed by the developers of each module.

2.3.3 Editing Google Documents

Findings: Editing Google Documents is inaccessible. This issue is non-trivial, and solving this problem will require work in Orca, WebKitGTK, and most likely Google Drive itself.

Recently an edit mode was added to GNOME Documents[12] so that Google Documents could be manipulated directly. When editing is enabled, the PDF view is replaced with a WebKitGTK view. As a result, the current level of accessibility found when editing Google Documents in Epiphany/Web is the same level of accessibility support found when editing Google Documents from within GNOME Documents. Unfortunately, that level of support is none.

Orca does have support for ARIA in Firefox. Adding similar suppport for ARIA in Epiphany/Web and improving Orca's overall ARIA support is both needed and planned. However, when navigating and editing Google Drive text documents and spreadsheets, there are no accessibility events seen in Epiphany/Web. Examining Firefox for comparison produced the same results. Additional investigation will be required in order to determine how much of the problem lies in the web rendering engines and how much lies in Google Drive's ARIA implementation. But given the complete lack of events seen in both browsers, our suspicion is that work in both areas will be required on top of the aforementioned improvements to Orca's ARIA support. As a result, we feel that Editing Google Documents is beyond the scope of this project.

2.4 PDFs Embedded in Epiphany/Web

Findings: Access to PDFs embedded in Epiphany/Web depends on Evince accessibility.

The ability to view PDFs directly in Epiphany/Web is not an accessibility-specific issue. However, because the plan is to make this possible,[13] we should be sure that it is accessible.

The patch attached to the bug was rebased and tested as part of the investigation conducted for this proposal, but PDFs were being shown in Evince rather than embedded in Epiphany/Web. Once that issue is resolved, this area can be formally assessed. However, because Evince's accessibility implementation is contained in libevview, embedded Evince documents should have the same level of accessibility support independent of the application embedding it. Because GNOME Documents also embeds Evince documents, it is our assumption that our assessment of that functionality applies to Epiphany/Web.

[12] https://bugzilla.gnome.org/show_bug.cgi?id=692102
[13] https://bugzilla.gnome.org/show_bug.cgi?id=689992

2.5 Document Support in GNOME's Accessibility Stack

Findings: ATK and AT-SPI2 do not require any modifications to make accessing PDFs possible. Orca may require some changes, but they are expected to be minimal in both number and size.

Orca provides access to document content through a combination of toolkit scripts, application scripts, and additional features for improved access to text:

- Structural navigation[14] provides navigation amongst elements according to type such as headings, tables and their cells, paragraphs, lists, landmarks, and form fields.
- Bookmarks[15] make it possible to temporarily store and optionally save one's location within documents based on the current accessible object.
- Text attribute support[16] facilitates obtaining font and other formatting information without the need to search for these details in application toolbars and/or dialog boxes.
- Label inference support[17] compensates for poor document authoring by attempting to infer what text is serving as the functional label when an accessible label has not been provided.
- Dynamic headers[18] allow the user to tell Orca what it should treat as the row and/or column headers as the user navigates within tables for which the author has not provided accessible headers.

The purpose of the toolkit scripts is to adjust for differences in ATK implementations; the purpose of the application scripts is to provide any additional, customized behavior required to further enhance Orca's support for a specific application. It would be surprising for neither to be needed to support the work planned for Evince. However, because one of the primary goals of this project is to provide an ATK implementation for Evince which adheres closely to both documented expectations and recognized best practices, it is our anticipation that Orca's support for accessing text and the document features mentioned above will work with Evince with a minimum number of changes required in Orca.

[14]https://help.gnome.org/users/orca/stable/howto_structural_navigation.html
[15]https://help.gnome.org/users/orca/stable/howto_bookmarks.html
[16]https://help.gnome.org/users/orca/stable/howto_text_attributes.html
[17]https://git.gnome.org/browse/orca/tree/src/orca/label_inference.py
[18]https://help.gnome.org/users/orca/stable/howto_tables.html#dynamic_headers

3 Proposed Work

In order to address as many of the issues found during our assessment as possible, the following areas of development are proposed:

3.1 Keyboard navigation

Modules involved: Evince, GNOME Documents

This area of work is small but critical, for without this in place the other required changes and additions will have little benefit.

During this project we will review, revise or redo, and commit the patch provided for Evince's lack of caret navigation.[1] In addition, we will add support to Evince to make objects within documents Tab focusable.[2] Lastly, we will provide fixes for the GNOME Documents keyboard navigation bugs described in our assessment.

3.2 Tagged PDF Support

Modules involved: Poppler, Evince

The amount of work required in this area is non-trivial, but its completion will bring about significant benefits in a number of areas of interest to GNOME.

With respect to accessibility, Evince will be able to expose document content as discrete accessible objects with a significant amount of structural and semantic information. This, in turn, will make it possible for Orca and other assistive technologies to provide truly compelling access to PDF document content.

But as this Adobe forum comment[3] explains, the benefits of tagged PDF support extend far beyond Accessibility and include:

- Reflow functionality
- Export to other applications with format, layout, font data, etc.
- Copy and paste to other applications with some fundamental retention of content format.

During this project, we will implement support in Poppler for at least ISO 32000-1:2008[4] section 14.8: Tagged PDF. In addition, we hope to implement support for PDF/UA-1 as specified in ISO 14289-1:2012.[5] But it should be noted that even without PDF/UA-1 support, accessibility of PDF content will be greatly improved.

[1] https://bugzilla.gnome.org/show_bug.cgi?id=638905
[2] https://bugzilla.gnome.org/show_bug.cgi?id=503706
[3] http://forums.adobe.com/message/2911759#2911759
[4] http://www.adobe.com/devnet/pdf/pdf_reference.html
[5] http://www.iso.org/iso/catalogue_detail.htm?csnumber=54564

3.3 Accessibility Support

Modules involved: Evince, GNOME Documents, Orca

This area of work builds upon the accessibility support that is already present in, or will be added to, Evince and GNOME Documents.

3.3.1 Evince

During this project we will review, revise or redo, and commit the patch provided for Evince's lack of accessible text attributes[6] and fix the selection-related bugs in Evince's AtkText implementation. We will also implement AtkObject, along with the appropriate properties and ATK interfaces for Evince's newly-exposed tagged elements.

In order to make form fields embedded in Evince documents more accessible, we will expose them via AtkText as "replaced objects" similar to what has been done in WebKitGTK, Gecko, and VCL. Having done so, we will add AtkHyperlink support for those characters, and provide sufficient AtkObject support for non-focused form fields[7] for Orca to be able to determine the form field's accessible role and location.

3.3.2 GNOME Documents

During this project we will give accessible names to all of the currently unnamed controls to which end users can navigate. We will remove ATK_STATE_SELECTABLE from the non-selectable objects to which end users can navigate. We will expose the displayed author information to assistive technologies. And we will discuss, and if appropriate make, the accessible signal/event changes needed for Orca to provide more compelling access to GNOME Documents than it currently does.

3.3.3 Orca

During this project we will make any additional changes or customizations needed and appropriate so that Orca provides more compelling access to Evince and GNOME Documents. This includes implementing handling for accessible tables which serve as menus, presenting the search results of GNOME Documents, and possibly creating a minimal custom script to handle any valid, but unhandled, differences in Evince's ATK implementation.

[6] `https://bugzilla.gnome.org/show_bug.cgi?id=639932`

[7] Form fields in Evince are not rendered as actual widgets until they gain focus, and these widgets are destroyed once the data has been entered or the field loses focus.

4 Timeline and Deliverables

This project has a projected duration of twelve weeks, to be spent as follows:

4.1 Week 1: Preparation

- File bugs for all of the issues identified during the assessment. DELIVERABLE
- Revise and commit the patch for text navigation. DELIVERABLE
- Revise and commit the patch for text attributes. DELIVERABLE
- Become familiar with the ISO specification(s) for Tagged PDFs.

4.2 Weeks 2-5: Tagged PDF Support, Text Selection

- Add support for tagged PDFs to Poppler core. DELIVERABLE
- Expose tagged PDFs in the Poppler GLib API. DELIVERABLE
- Analyze what changes need to be made in Evince to use the new Poppler support.
- Implement the required changes in Evince. DELIVERABLE
- Fix the bugs in Evince's AtkText selection implementation. DELIVERABLE

4.3 Weeks 6-9: Expose Information to Assistive Technologies

- Implement AtkObject for all available tagged elements. DELIVERABLE
- Implement the AtkTable interface for tagged table elements. DELIVERABLE
- Fix the issues identified related to GNOME Documents. DELIVERABLE

4.4 Weeks 10-12: Form Field Accessibility

- Make Evince form fields Tab focusable. DELIVERABLE
- Represent form fields within AtkText using "replacement characters." DELIVERABLE
- Implement AtkHyperlink support for "replacement characters." DELIVERABLE
- Implement AtkObject support for non-focused fields sufficient to expose their role and extents. DELIVERABLE

Final deliverable: Summary of work completed along with detailed reassessment of GNOME's PDF accessibility.

Please note: The timeline and deliverables described above may be affected by unexpected complications. Given our thorough initial assessment and our positions relative to the modules in which the work is to be done, we are not anticipating very many difficulties. The two areas in which there is some uncertainty are: Exactly how much work will be required to implement the tagged PDF specification in Poppler and Evince, and how best to implement AtkObject support for objects which do not exist as such (namely the non-focused form fields).

5 Developers

Igalia plans to allocate, on a part-time basis, three developers whose expertise and experience cover all of the areas listed in the "Proposed Work" section of this response. These developers will do the bulk of the development, and will directly oversee the work of any other developers from Igalia whose contributions are required.

5.1 Carlos García Campos

Carlos[1] has been involved in the free software community since 2002, doing development and leading projects in both GNOME and freedesktop.org. He has over five years of experience in PDF technologies, is the maintainer of Evince and libgxps, and is one of the maintainers of Poppler. In addition to his PDF development, Carlos is a WebKit reviewer currently focused on WebKit2 and also a signifant contributor to Epiphany/Web.

5.2 Alejandro Piñeiro Iglesias

Alejandro[2] has been a free software developer since 2004. While his experience includes a variety of GNOME and freedesktop.org projects, his focus since 2007 has been Accessibility. Alejandro started the Accessibility Group at Igalia, leads the GNOME Accessibility Team, and serves as a member of the GNOME Release Team. He is the maintainer of ATK, HAIL (Hildon Accessibility Implementation Library), and Cally (Clutter's ATK implementation). He implemented accessibility support in GNOME Shell and continues to contribute to other GNOME Accessibility modules and implementations including Gtk+, WebKitGTK, AT-SPI2, and Orca.

5.3 Joanmarie Diggs

Joanmarie[3] has worked in the field of Accessibility for over 16 years: From 1996 until 2011, she was an Assistive Technology Specialist, Orientation & Mobility Specialist, and Vision Rehabilitation Therapist. For the last five of those years, Joanmarie was also a volunteer developer of the Orca screen reader and an active member of the GNOME Accessibility Team. In 2011 Joanmarie joined Igalia's Accessibility Group where she works full-time on accessibility solutions for the free desktop. She continues to be the maintainer and primary developer of Orca, is a WebKit accessibility developer, and contributes to other GNOME Accessibility modules including ATK, AT-SPI2, and Accerciser.

[1] http://www.ohloh.net/accounts/carlosgc
[2] http://www.ohloh.net/accounts/infapi00
[3] http://www.ohloh.net/accounts/joanmarie

6 References

The following references provide an overview of the relevant work done by the developers listed in the previous section. Additional details will be provided upon request.

6.1 Document-Related Contributions

- https://git.gnome.org/browse/evince/log/?qt=author&q=carlosgc
- https://git.gnome.org/browse/libgxps/log/?qt=author&q=carlosgc
- http://cgit.freedesktop.org/poppler/poppler/log/?qt=author&q=carlosgc
- https://git.gnome.org/browse/epiphany/log/?qt=author&q=carlosgc
- http://trac.webkit.org/search?changeset=on&q=carlosgc
- http://trac.webkit.org/search?changeset=on&q=diggs
- http://trac.webkit.org/search?changeset=on&q=apinheiro

6.2 Accessibility-Related Contributions

- https://git.gnome.org/browse/gnome-shell/log/?qt=author&q=apinheiro
- https://git.gnome.org/browse/gtk+/log/?qt=author&q=apinheiro
- https://git.gnome.org/browse/clutter/log/?qt=author&q=apinheiro
- http://maemo.gitorious.org/hail
- https://git.gnome.org/browse/atk/log/?qt=author&q=apinheiro
- https://git.gnome.org/browse/atk/log/?qt=author&q=joanied
- https://git.gnome.org/browse/at-spi2-atk/log/?qt=author&q=apinheiro
- https://git.gnome.org/browse/at-spi2-atk/log/?qt=author&q=joanied
- https://git.gnome.org/browse/pyatspi2/log/?qt=author&q=joanied
- https://git.gnome.org/browse/orca/log/?qt=author&q=joanied
- https://git.gnome.org/browse/orca/log/?qt=author&q=apinheiro
- https://git.gnome.org/browse/accerciser/log/?qt=author&q=joanied

7 Conditions and Contacts

7.1 Conditions

As indicated in the call for bids,[1] the total cost of this project is $30,000. Regarding payments, Igalia is flexible and open to what is most convenient for the GNOME Foundation.

7.2 Sales Contact

Juan José Sánchez Penas <jjsanchez@igalia.com>
Telephone: +34 981 91 39 91

7.3 Technical Contacts

Joanmarie Diggs <jdiggs@igalia.com>
Telephone: +1 603 882 0393

Alejandro Piñeiro Iglesias <apinheiro@igalia.com>
Telephone: +34 981 91 39 91

Carlos García Campos <cgarcia@igalia.com>
Telephone: +34 981 91 39 91

[1] http://www.gnome.org/news/2013/02/call-for-bids-for-gnome-accessibility-work/