



The Information Company

Accessibility Report

My Support Portal

Rachele DiTullio

Sr. IT Software Engineer, EBS Web Team — OpenText

31 January 2020

Table of Contents

Executive Summary **3**

Methodology **4**

 Scope 4

 Support baseline..... 5

 Automated testing 5

 Assistive technologies (AT) 6

 Heuristics..... 6

 Considerations..... 7

Findings **8**

 Sitewide concerns 8

 Layout..... 12

 Tickets home 15

 View a ticket 18

 Open a new technical ticket 19

 Product activation..... 21

 View an account..... 24

Conclusion..... **26**

 Further Learning..... 26

 Other Sites..... 27

References **28**

Executive Summary

OpenText is required to comply with the [Accessibility for Ontarians with Disabilities Act, 2005 \(AODA\)](#) as a private organization with 50+ employees based in Ontario. These regulations require “new and significantly refreshed public websites” meet WCAG 2.0 guidelines (Government of Ontario, Canada, 2020).

Beginning January 1, 2014: new public websites, significantly refreshed websites and any web content posted after January 1, 2012 must meet Web Content Accessibility Guidelines (WCAG) 2.0 Level A

Beginning January 1, 2021: all public websites and web content posted after January 1, 2012 must meet WCAG 2.0 Level AA

Accessibility is a way of thinking: it’s habit and process, not a finish line. Good accessibility is intentional, starts with design and is everyone’s responsibility. Much in the way we had to change our thinking and design approaches to achieve responsive layouts for mobile devices, we must now do the same with accessibility.

This report discusses some of the accessibility issues with the current state of the **My Support Portal** and reviews the tickets grid, an individual ticket, opening a new technical ticket, product activation and accounts.

Common problems include:

- Content inaccessible to people using keyboards
- Content inaccessible to people using screen readers and other assistive technologies (AT)
- Content inaccessible to people with low vision due who need support for text resizing as well as good contrast between text and background colors
- Lack of ARIA (Accessible Rich Internet Applications) support for custom widgets that depend on JavaScript to function such as data grids, site navigation and modal dialogs
- Lack of text alternatives for icons and controls
- Generic link text

As IT continues to maintain the My Support Portal, the design and development planning must reference and conform with WCAG 2.0 guidelines to be compliant with Canadian law. The following report outlines the heuristics used when evaluating user tasks and shows through real examples some of the many accessibility challenges of using this web application.

Methodology

The report references the Web Content Accessibility Guidelines (**WCAG**) **2.0 Level AA**.

WCAG 2.0 covers “... a wide range of recommendations for making Web content more accessible. Following these guidelines will make content accessible to a wider range of people with disabilities, including blindness and low vision, deafness and hearing loss, learning disabilities, cognitive limitations, limited movement, speech disabilities, photosensitivity and combinations of these. Following these guidelines will also often make your Web content more usable to users in general.” (World Wide Web Consortium, 2008)

Scope

The scope of this review is limited to pages within the My Support Portal, located at <https://support.opentext.com/portal/site/css>*

The pages assessed in this report are listed in the table below with usages statistics from Google Analytics, 1 July through 31 December, 2019 (Google Analytics, 2020). During this time period, there were **816,153 total pageviews**.

Five areas comprise this accessibility review with multiple pages and screens evaluated to demonstrate how people with certain disabilities might experience a task such as opening a new ticket.

Page Name	URL	Pageviews	% of all Pageviews
<i>Tickets Home</i>	https://support.opentext.com/portal/site/css?customview=ticketshome	113,751	12.29%
<i>View a Ticket</i>	https://support.opentext.com/portal/site/css?ticketId=	266,931	39.65%
<i>New Technical Ticket</i>	https://support.opentext.com/portal/site/css?customView=newTicketTechnical	50,288	6.16%
<i>Product Activation</i>	https://support.opentext.com/portal/site/css?customview=activationshome	24,969	3.06%

Page Name	URL	Pageviews	% of all Pageviews
View an Account	https://support.opentext.com/portals/site/css?customview=accounts/home	24,481	3.00%

Support baseline

The review was conducted on a laptop running Windows 10 using these web browsers:

- Chrome (version 79)
- Firefox (version 72)
- Internet Explorer (version 11)

Additional operating system and browser combinations were not tested, including robust testing with mobile devices.

Responsive design and other mobile considerations were tested by adjusting the desktop browser viewport.

Input devices were an integrated touch QWERTY keyboard and a wireless, five-button mouse.

Automated testing

The following tools were used for initial **automated** accessibility testing for issues like document outline, color contrast and text alternatives:

- [Lighthouse accessibility audit – Chrome developer tools](#)
- [axe for Web – browser extension for Chrome and Firefox](#)

Automated testing tools can uncover **only about 30% of potential issues**. Each tested page includes:

- The page’s accessibility score as rated by Lighthouse
- The number of accessibility problems found with Axe

While automated tests can determine if particular elements are present, e.g. whether images have alt text, tools cannot always determine if the implementation satisfies the WCAG requirement.

Example: A photo of a dog with `alt="photo"`

The alt text fails to describe the content of the image so it does not meet the success criterion for a text alternative, even though an alt attribute is present.

All automated testing is followed up with manual testing.

Assistive technologies (AT)

Each page was evaluated through manual testing with the following AT:

- [NVDA screen reader](#)
- Windows high contrast mode (Windows Key+U) and IE
- Zoom text 200% (Firefox)

Heuristics

WCAG is organized by four principles and 12 guidelines for each conformance level (A, AA and AAA). The AODA requires conformance to **36 success criteria**:

Principle	Guidelines	Success Criteria
Perceivable	<p>Information and user interface components must be presentable to users in ways they can perceive.</p> <p>1.1. Text alternatives: Provide text alternatives for non-text content.</p> <p>1.2 Time-based media: Provide captions and other alternatives for multimedia.</p> <p>1.3 Adaptable: Create content that can be presented in different ways, including by assistive technologies, without losing meaning.</p> <p>1.4 Distinguishable: Make it easier for users to see and hear content.</p>	<p>Level A – 9</p> <p>Level AA – 3</p> <p>(Criteria for live captions and pre-recorded audio descriptions are excluded by the AODA.)</p>
Operable	<p>User interface components and navigation must be operable.</p> <p>2.1 Keyboard accessible: Make all functionality available from a keyboard.</p> <p>2.2 Enough time: Give users enough time to read and use content.</p> <p>2.3 Seizures and physical reactions: Do not use content that causes seizures or physical reactions.</p> <p>2.4 Navigable: Help users navigate and find content.</p>	<p>Level A – 9</p> <p>Level AA – 3</p>

Principle	Guidelines	Success Criteria
Understandable	<p>Information and the operation of user interface must be understandable.</p> <p>3.1 Readable: Make text readable and understandable.</p> <p>3.2 Predictable: Make content appear and operate in predictable ways.</p> <p>3.3 Input assistance: Help users avoid and correct mistakes.</p>	<p>Level A – 5</p> <p>Level AA – 5</p>
Robust	<p>Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies.</p> <p>4.1 Compatible: Maximize compatibility with current and future user tools.</p>	<p>Level A – 2</p>

Reference [How to Meet WCAG 2](#) for explanations of the guidelines and success criteria.

Not all guidelines are applicable to every page, e.g. if a page does not include video, then it does not need to be tested for captioning.

Considerations

- 1) This review process did not include people with disabilities evaluating the web application. Future audits should include representative users utilizing a variety of assistive technologies.
- 2) The web application does not employ responsive design which is not required by WCAG 2.0 but was added in WCAG 2.1 as **1.4.10 Reflow** along with **1.3.4 Orientation**: Content does not restrict its view and operation to a single display orientation, such as portrait or landscape. Laws are continually changing; designing for WCAG 2.1 now will help future-proof new work.
- 3) Future audits should be evaluated using mobile devices with iOS/VoiceOver and Android /TalkBack screen readers.
- 4) WCAG 2.0 *conformance* claims cannot be made for entire websites based on the evaluation of a select set of web pages and functionality. **Conformance** is “[s]atisfying all the requirements of a given standard, guideline or specification” (WCAG 2.0 Evaluation Methodology Task Force, 2014).
- 5) This assessment is not an exhaustive list of any government-required accommodations, such as Section 508 of the United States Rehabilitation Act of 1973 and the AODA.

Findings

The results of each page or screen reviewed are **cumulative**. If an issue is something that affects all pages, e.g. the default link color has poor color contrast with the page background, every instance is not documented as this is a sitewide issue that does not bear repeating.

Sitewide concerns

These are severe issues that have the potential to affect every page on the site through CSS styling rules, third-party libraries that are inaccessible and inclusion of non-HTML content.

Focus indicator

Any keyboard operable user interface must have a mode of operation where the keyboard focus indicator is visible.

There are several areas where keyboard focus is very hard to see or is missing altogether. An example is all the links in the global navigation, which have no visible focus indicator.

A general rule of thumb is to ensure that `:hover` rules also apply to `:focus`. Whatever happens when a user hovers over an element should likely occur when it has focus too.

Button and link purpose (in context)

The purpose of each button control or link must be able to be determined from the link text alone or from the link text together with its programmatically determined link context. (See: [Links must have discernable text](#))

All buttons and links must have text that can be accessed by a screen reader. This often affects elements used for controls and links that use CSS background images or pseudo selectors to inject content which is not read by screen readers. The control below is from the Tickets grid has no label:



```
<button type="button" wiid="C230" id="ext-gen43" class=" x-btn-text x-tbar-page-next">&nbsp;&nbsp;&nbsp;</button>
```

Link text should accurately describe the link target. Using generic labels like “learn more” and “click here” does not provide context to screen reader users who are tabbing through the focusable page elements.

```
<p class="ia-content"><span class="ia-title">Maintenance Notice - </span><a href="https://knowledge.opentext.com/knowledge/cs.dll?func=11&objId=77457947&objAction=ViewNews&viewType=1" target="_blank">January 24, 2020.</a></p>
```

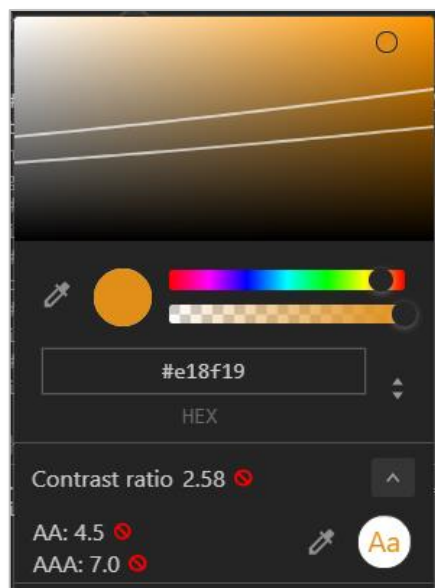

In the above example, the text “Maintenance Notice” isn’t linked, only the date is. People using assistive technology are not provided the proper link context.

Color contrast

Most text must have a contrast ratio of at least **4.5:1**. Large text can have a contrast of 3:1. Color contrast can be checked from the developer tools console in Chrome and a new color choice can be tested directly. (See: [Text elements must have sufficient color contrast against the background](#))

The below example shows the orange color to indicate a “Non-production” license type which has a contrast ratio of just **2.58** on a white page background.

SUID:52318-12381 (4 LICENSES)			
Product Line	Version	License Type	Status
Content_Server	10.5.x	Non-production	Active



Other areas with color contrast issues:

- Orange link hover text in product activations
- Username in header (low opacity)
- Footer links
- Copyright text

Forms

Forms on the My Support portal contain several issues for screen reader accessibility, particularly with required fields and form validation.

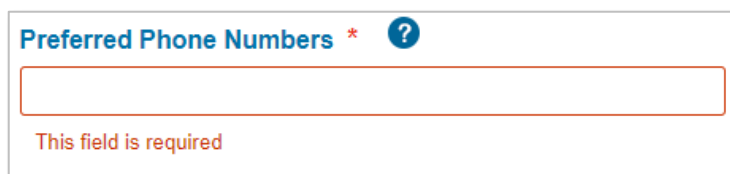
- 1) The form fields should use the `required` attribute to mark them up semantically for screen readers as the asterisk is a visual indicator of required fields.

```
<input id="prefPhoneNum" name="preferredPhoneNumbers" type="text" value="8085216227 x11" size="60" class="x-form-text x-form-field x-form-invalid" required>
```

Using this technique, the screen reader will announce: *Preferred Phone Numbers*, **edit required**

See the [Forms Concepts > Validating Input tutorial](#).

- 2) Disabling form submit buttons is confusing for people with low vision or who are blind. For someone using a screen reader, they simply won't be able to find a disable button because it cannot gain keyboard focus. Additional help text or not disabling the button is needed.
- 3) The form field validation is insufficient for screen reader users because the error text is not available to the screen reader while it is in forms mode.



Preferred Phone Numbers * ?

This field is required

Associate help text with a form field using the `aria-describedby` attribute.

```
<input aria-describedby="preferredPhonneInvalid" id="preferredPhoneNumbers" name="preferredPhoneNumbers" type="text">  
<div id="preferredPhonneInvalid">This field is required</div>
```

See the [Forms Concepts > User Notifications tutorial](#).

PDF content

Web content served in PDF format is also subject to WCAG success criteria. Just as with HTML, PDF content must

- Be accessible to screen readers and keyboard users
- Maintain a proper tabbing order through the content
- Use document title, language, headings, tables, etc. to tag the content semantically
- Provide alternative text for images
- Include links for linked text

The ticket PDF export, for example, does not tag any of the content such as headings, that would make it easy to read and navigate.

In Acrobat go to **Tools > Accessibility > Reading Order** to see issues.

Resources

- [Creating accessible PDFs](#)
- [Create and verify PDF accessibility \(Acrobat Pro\)](#)
- [PDF Techniques for WCAG 2.0](#)
- [Checklist of Standards, Techniques & Tests for Accessible PDFs](#)

High contrast mode

Some users employ OS settings or browser extensions to view screens in high contrast mode. This site loses some of its context in this mode because it relies on background images:

- The OpenText logo is a background image and is not visible in high contrast mode. The same effect would be seen if the logo were an inline image with black text and transparent background.
- Any dark text that relies on a background image for contrast could become invisible
- Controls that rely on background images, become invisible, like those in the Tickets grid for navigating as well as exporting to PDF

Maintenance Notice - [January 24, 2020](#) Rachele DiTullio

Home Products Knowledge Base Tickets Forums

Resources My Tickets My Accounts Services & Programs Partner Help

Can we help you find something?

Tickets

Displaying tickets for accounts: All Accounts

Display State Status Ticket Type Product Line

Displaying tickets 1 - 25 of 386 View All Page 1 of 16

Ticket #	Subject	Created	Last Modified	Priority	Status	Contact	Product Line	Account	System	Export to PDF
4363236	RECOMMIND - On Demand - DLA - Fairchild - Account Activatio...	1/24/2020 11:...	1/24/2020 11:...	2	Action-O...	Louis Ne...	Actuate	Open Text Corp H...		
4363221	Canon: 16.3.7: Export Not working	1/24/2020 10:...	1/24/2020 12:...	2	Action-O...	Chandra...	Media Manag...	Open Text Corp H...	Media M...	
4362709	Recommind Accelerate - Reset/Unlock User password/account	1/23/2020 8:1...	1/24/2020 12:...	3	Action-C...	Angello...	Recommind	Open Text Corp H...		
4362605	Download with Relationship "has video clips" Doesn't Work	1/23/2020 6:0...	1/23/2020 6:5...	2	Action-O...	Simon Z...	Media Manag...	Open Text Corp H...	Open Te...	

Layout

The report first covers issues that affect every page on the site. *These issues are not repeated in the reviews of specific pages.*

This site uses some `<table>` elements for layout which is confusing to people navigating the site using AT. All content should be marked up using semantic HTML5.

Header

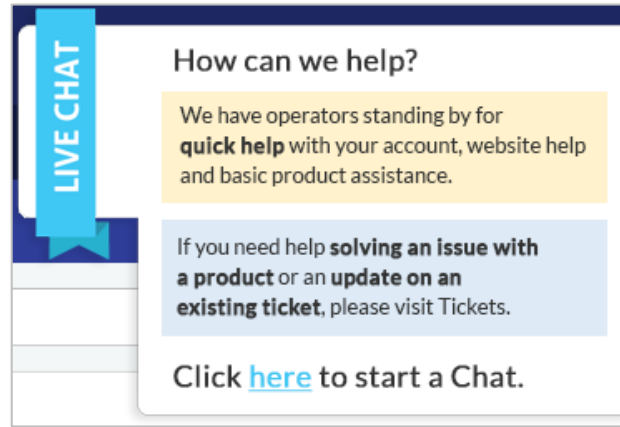
- 1) The `<html>` element of all pages must define the language of the content. If page content language changes based on user preferences, the `lang` attribute must be updated. When the correct language is not specified, AT cannot accurately communicate the contents.
- 2) The global navigation menus should be located within the `<header>` ARIA landmark element. (See: [All page content must be contained by landmarks.](#))
- 3) The My Support Portal pages load with keyboard focus in the search field. This is confusing for AT users who expect focus to be at the beginning of the document.
- 4) There is no "skip navigation" link available for keyboard users to skip tabbing through all the navigation elements of the page to reach the main content area of the page. See www.opentext.com for an example of how this works. The "skip to main content" link is the first thing that gains keyboard focus when a user starts navigating the site. It is invisible to most users.



- 5) Every page in the site has a `<title>` of "My Support." Each page requires a unique page title that clearly indicates the main contents of the page, e.g. the Tickets page should have a title of "My Support **Tickets**" or similar.
- 6) The two search `<input>` elements do not have `<label>` elements for AT users. The text "Can we help you find something?" must be programmatically associated with the search fields.
- 7) The source order for the search widget is incorrect. Keyboard users tabbing through the site first access the "Text search" dropdown, then the "Search" button, then the text `<input>` for the search query.
- 8) I was not able to access the Live Chat feature with a keyboard. Further, this widget is a giant image with text but provides no text alternative which results in an empty link.

```

```



Navigation

The navigation elements are not marked up semantically as `<nav>` elements. When content is marked up using HTML5 elements or ARIA landmarks, screen reader users can navigate pages more easily. Distinguish [multiple uses of the navigation landmark](#) on a page by giving each a unique label.

Example:

```
<nav aria-label="global">
<ul id="selectedResourceNav" class="menu-horizontal" style="border-color:
#2E3D98 !important; top: 60px;">
...
</ul>
</nav>
```

- 1) The site's main navigational links are not accessible with a keyboard because they rely on JavaScript that doesn't account for keyboard focus or keypresses. While users can tab through the main items, they cannot access any of the links in the submenus without a mouse.
- 2) There is no clear and obvious **focus indicator** for keyboard users to know which element has focus on the page. This makes navigating data grids impossible.

Main content

All main page content (that is not already in headers, footers, asides) must be contained within a `<main>` element. This is also the focus of the "skip navigation" link. (See: [All page content must be contained by landmarks.](#))

Footer

- 1) The footer link color does not provide a contrast of at least **4.5:1**. The gray text (#9c9c9c) has a contrast of **2.52** against the light gray background (#f5f5f5).



- 2) The footer link hover text color does not provide a contrast of at least **4.5:1**. The white text (#fff) has a contrast of **1.09** against the light gray background (#f5f5f5).



3) Links do not have a discernible name – 5 instances

Link text (and alternate text for images, when used as links) that is discernible, unique, and focusable improves the navigation experience for screen reader users. (See: [Links must have discernible text](#))

The five social media links use icons without providing a text alternative.

```
<li class="twitter"><a target="_blank"
href="https://twitter.com/OpenText">&nbsp;</a></li>
<li class="google-plus"><a target="_blank"
href="https://plus.google.com/u/0/b/108211336773562179239/+Opentext2020/
posts">&nbsp;</a></li>
<li class="linked-in"><a target="_blank"
href="https://www.linkedin.com/company/opentext">&nbsp;</a></li>
<li class="youtube"><a target="_blank"
href="https://www.youtube.com/user/opentextcorp">&nbsp;</a></li>
<li class="flipboard"><a target="_blank"
href="https://flipboard.com/@OpenText">&nbsp;</a></li>
```

Tickets home

Accessibility score: 48/100

Automated issues: 64

Buttons do not have an accessible name – 5 instances

When a button doesn't have an accessible name, screen readers announce it as "button", making it unusable for users who rely on screen readers. (See: [Buttons must have discernible text](#))

Several button controls in the ticket grid have no text alternatives: first page, previous page, next page, last page and refresh.

```
<button type="button" wiid="C280" id="ext-gen49" class=" x-btn-text x-tbar-
loading">&nbsp;</button>
```



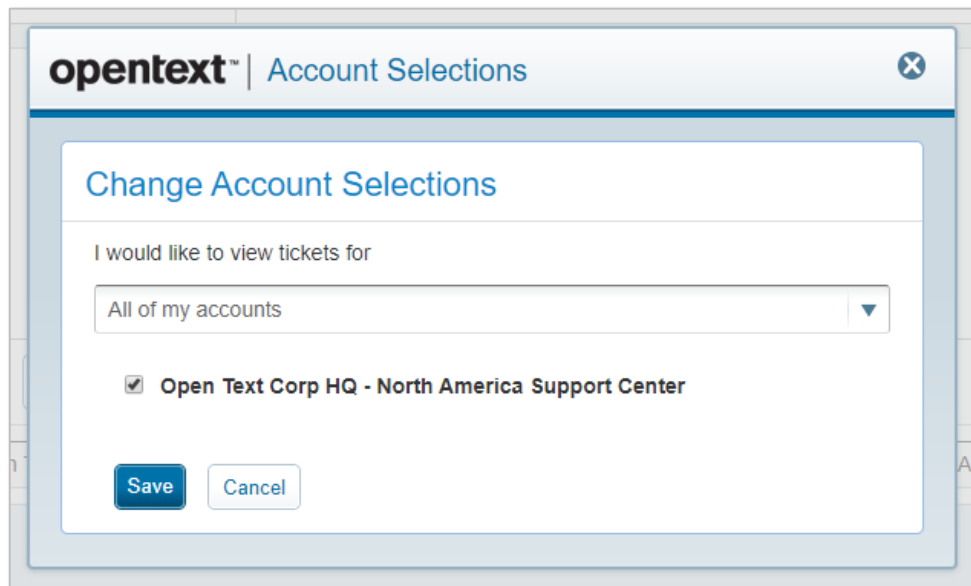
This issue and others with the data grid affect **every instance** where this data grid pattern is used in the web application.

Manual testing

User Story: As a keyboard-only user, I want to be able to change my account selections so that I can review my open tickets for a specific account.

Focus does not move to "Change Account Selections" modal window

When a keyboard user attempts to interactive with the modal window, they cannot because the keyboard focus remains on the page content behind it. Always move focus to the modal dialog and keep it there until the user wishes to exit.



And while this modal dialog has a "close" button, it has no text alternative label and is not a keyboard-focusable element because it is marked up as a `<div>`.

```
<div class="x-tool x-tool-close" wiid="N2359" id="ext-gen111">&nbsp;  </div>
```

These issues affect every instance where this modal window pattern is used in the web application.

User Story: *As a screen reader user, I want to be able to be able to navigate my list of tickets so I can find the right one to open.*

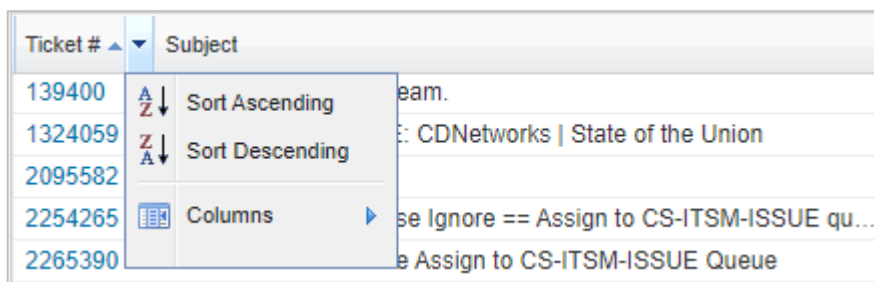
Tickets grid does not properly associate column headers with the contents of table rows

The tickets grid is made up of several nested tables instead of properly marking up with individual rows and columns in a single data grid that can be navigated with AT like a screen reader.

- Column names are not announced by a screen reader before each data cell, e.g. a screen reader should announce

row 2 column three, Created, 1/28/2020

- The number of ticket rows is not announced properly because the data cells are in nested tables, e.g. the screen reader announces the data grid has just two rows no matter the number of tickets available; this nesting also prevents navigating the table data with a keyboard
- There is no indication that a column can be sorted
- There is no user feedback when the list of tickets is updated due to a user action (sighted users can see the "Loading tickets" overlay and the number of tickets change)
- Keyboard users cannot access any of the data grid's sort controls on the columns as they require a mouse



These issues affect every instance where this data grid pattern is used in the web application.

View a ticket

Accessibility score: 48/100	Automated issues: 64
------------------------------------	-----------------------------

Form elements must have labels – 3 instances

Several form fields have mismatches between the `<label>` for value and the `<select>` or `<input>` ID value. This prevents them from being programmatically associated.

```
<div class="fieldData middle">
  <label for="priority">Priority</label>
  <select id="prioritySelect" name="priority">...</select>
</div>
```

Manual testing

User Story: *As a keyboard-only user, I want to be able to be able to update my preferred contact method to "phone" so that the CSR will call me.*

Keyboard-only users cannot navigate to any of the collapsed sections of the ticket screen because the clickable areas are not marked up as keyboard-focusable elements:

```
<div class="x-panel-header x-unselectable" wiid="N944" id="ext-gen166">
  <div class="x-tool x-tool-toggle" wiid="N948" id="ext-gen169">&nbsp;</div>
  <span class="x-panel-header-text" wiid="N947" id="ext-gen170">Contact
  Information</span>
</div>
```

These collapsed sections can't be expanded by keyboard-only users so they cannot update contact information, see ticket details, assignments, escalation details or development details. They are restricted to interacting with the "Event Details" area which is expanded by default.

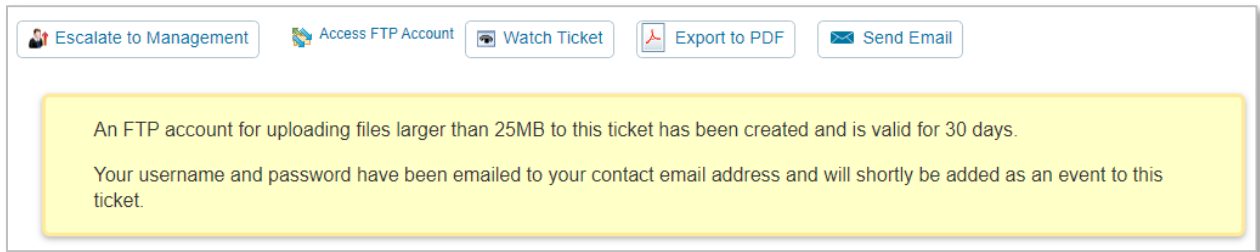
This interaction issue affects every instance where the accordion pattern is used in the web application.

User Story: *As a screen reader user, I want to be able to request an FTP account so I can upload a large file.*

FTP account information is not communicated to screen readers

When screen reader users access the "Request FTP Account" link, they are not provided with any feedback. Sighted users can see two things happen:

- The "Request FTP Account" link changes to "Access FTP Account"
- A message about using the FTP account appears



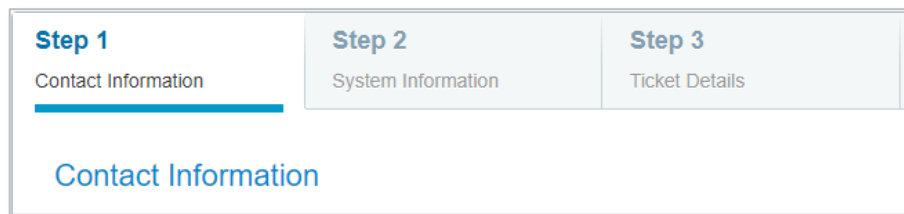
Neither of these changes is communicated screen reader users nor is the alert message in programatically close proximity to the new "Access FTP Account." To be accessible, focus must move to the message and instructions must be provided on how to find the "Access FTP Account" link.

Open a new technical ticket

Accessibility score: 60/100	Automated issues: 52
------------------------------------	-----------------------------

1) **Background and foreground colors do not have sufficient contrast ratio**

The light blue color of the "Contact Information" heading (#08C) has a contrast ratio of just **3.89** against the white background. Additionally, the faded opacity for displaying that tickets have three steps do not meet contrast ratios for someone with low vision.



2) **Form elements must have labels**

Some form fields have mismatches between the `<label>` *for* value and the `<select>` or `<input>` *ID* value. This prevents them from being programmatically associated.

```
<label for="preferredEmailAddresses">Preferred Email Addresses <span class="required" title="required">*</span></label>
<input id="prefEmailAddr" name="preferredEmailAddresses" type="text" value="" size="60" class="x-form-text x-form-field" title="">
```

Manual testing

User Story: *As a screen reader user, I want to be able to add multiple preferred phone numbers to my account so that I can be reached at work or on my mobile.*

This form is difficult to navigate because of the multiple headings that don't contain content. All three "steps" are marked up like empty sections using both `<h2>` and `<h3>` elements without any context for non-sighted users about the form being a three-step process with multiple screens.

Navigating to the "Preferred Phone Numbers" field is tricky given the form issues noted above. This is what the screen reader announces:

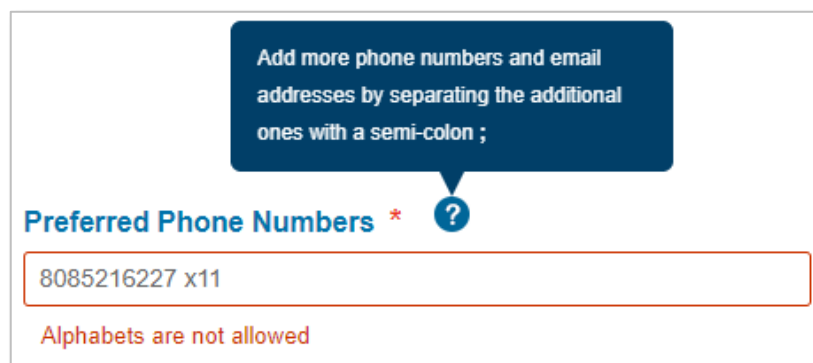
Preferred Phone Numbers

** [star]*

edit

8085216227 x11

Alphabets are not allowed



The help text about adding additional phone numbers separated with a semi-colon is not available to the screen reader because the help icon is not marked up as a keyboard-focusable element nor is the appropriate JavaScript used for it to work as a custom widget.

```
<span class="questionMark" title=""></span>
```

The error text is difficult to understand and does not explain how to fix the error. The default phone number that loaded from the system has an "x" character already. Saying "Alphabets are not allowed" when the user didn't enter the data makes it very hard to understand what to fix, especially when the "Next Step" button is disabled.

User Story: *As a keyboard user, I want to be able to complete the new ticket form so that I can log my support issue.*

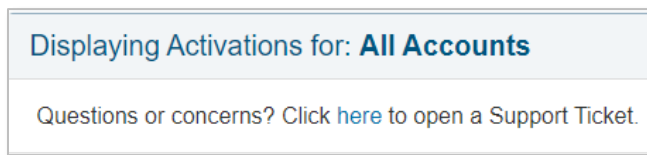
Keyboard users must be able to see where the focus is on the page. It's nearly impossible to tell where you are in filling out these forms where certain fields require other fields be filled out first.

When a user goes to the screen for Step 2, keyboard focus remains on the "Next Step" button, making it very hard for keyboard and screen reader users to know where they are in the page to continue. Keyboard users have to use Shift+Tab keys to move the focus back to the first form field.

Product activation

Accessibility score: 64/100	Automated issues: 170
------------------------------------	------------------------------

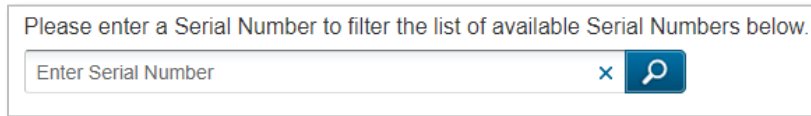
1) **Links do not have a discernable name**



Link the text "Open a Support Ticket" and avoid using "click here."

2) **Form elements must have labels**

The Guidance serial number search box uses placeholder text instead of a `<label>` element. This prevents users from knowing what information to enter into the field and then obfuscates the "label" once text is entered, causing cognitive load. Also, the light gray placeholder text often does not satisfy color contrast requirements.



Manual testing

User Story: *As a screen reader user, I want to be able to download my Content Server product activation key so I can get our new environment up and running.*

This page is extremely cumbersome to navigate with a screen reader. There are two major issues:

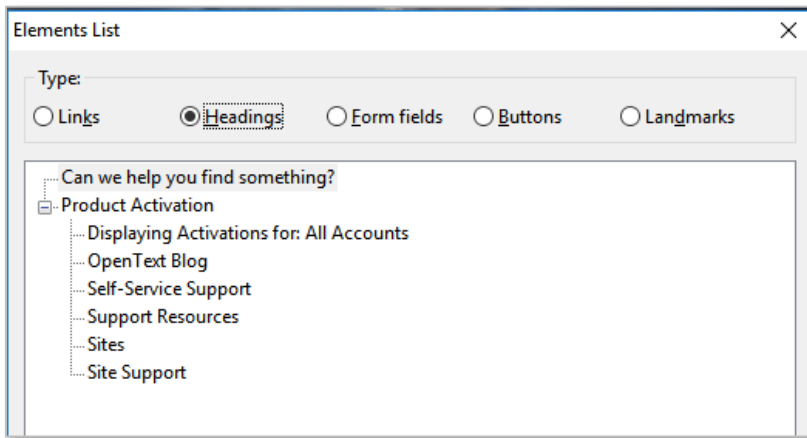
- 1) Screen reader users don't just read through the entire contents of a page; they are able to use keyboard shortcuts to do things like hear a list of all the links on the page or a list of headings—this allows someone listening to the content to "skim" the page without having to go through every, single word.





Unfortunately, the headings on this page are visual only and not marked up with proper `<h1-h6>` tags that would allow a screen reader user to skip over Actuate and go directly to the Content Server section. Additionally, most of the sections are expanded by default and the "collapse/expand section" buttons are not programmatically linked to the section name each button is activating.

```
<div id="gridTopHeader2" class="gridTopHeader">Actuate<table id="ext-comp-1051" cellpadding="0" class="x-btn hideShowButton x-btn-noicon"
```

```
wiid="N1068" style="width: auto; "><tbody class="x-btn-small x-btn-icon-small-left" wiid="C1069"><tr wiid="C1070"><td class="x-btn-tl" wiid="C1071"><i wiid="C1072">&nbsp;</i></td><td class="x-btn-tc" wiid="C1073"></td><td class="x-btn-tr" wiid="C1074"><i wiid="C1075">&nbsp;</i></td></tr><tr wiid="C1076"><td class="x-btn-ml" wiid="C1077"><i wiid="C1078">&nbsp;</i></td><td class="x-btn-mc" wiid="C1079"><em class="" unselectable="on" wiid="C1080"><button type="button" wiid="C1081" id="ext-gen216" class=" x-btn-text">Collapse Section</button></em></td><td class="x-btn-mr" wiid="C1082"><i wiid="C1083">&nbsp;</i></td></tr><tr wiid="C1084"><td class="x-btn-bl" wiid="C1085"><i wiid="C1086">&nbsp;</i></td><td class="x-btn-bc" wiid="C1087"></td><td class="x-btn-br" wiid="C1088"><i wiid="C1089">&nbsp;</i></td></tr></tbody></table></div>
```



2) As with the data grids in other parts of this web application the overuse of <table> elements results in great difficult navigating the individual licenses. For tables to be useful, they must communicate the column name before the data cell contents.

SUID:52318-12381 (4 LICENSES)							
Product Line	Version	License Type	Status	Expiration	Quantity	Activated	Actions
Content_Server	10.5.x	Non-production	Active	N/A	10	10	 

In this example, we see what looks like a heading above a table with a header row and a data row, comprised of nine columns. That is not how this information is coded, though.

- The heading text should be marked up as a table <caption> or other programmatic way to make it describe the contents of the table.

- The header row is actually a separate table using CSS to make it look like column headers but functionally, they do nothing.
- Since the table row of data cells is not programmatically linked to the column headers, a screen reader announces only the contents of a cell without the context of the header, e.g.

*table with two rows and nine columns
row two column three license type, non-production*

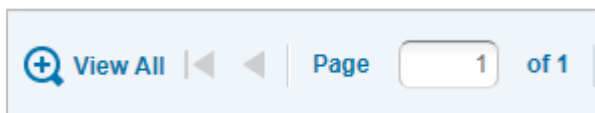
- When tabular data is marked up properly, screen readers can enter "tables" mode which allows users to navigate within the table itself using arrow keys.

View an account

Accessibility score: 48/100	Automated issues: 60
------------------------------------	-----------------------------

1) Form elements must have labels

As with other parts of the web application that use the data grid pattern, the field to select which page of results to view does not have a proper `<label>` element.



The entire menu bar is comprised of complicated nested tables. The words "Page" and "of 1" must be announced to the user when focus goes into the pagination field. Instead, these are marked up as `<div>` elements within `<td>` elements which separates them programmatically.

```
<td>
<div class="xtb-text" id="ext-comp-1014">Page</div>
</td>
<td>
<input type="text" size="20" autocomplete="off" id="ext-comp-1006"
wiid="N195" class="x-form-text x-form-field x-form-num-field x-tbar-
page-number">
</td>
<td>
<div class="xtb-text" id="ext-comp-1007">of 1</div>
</td>
```


Manual testing

User Story: As a screen reader user, I want to be able to "watch" all the tickets on my account so that I am alerted by email about ticket updates.

The tabbing order of the Account Information section is confusing and does not follow the logical reading order which makes it difficult for people who cannot see the page to find what they want. When regular text content is mixed in with `<form>` content, it is not read by a screen reader that is in "forms" mode unless it is programtically associated with form elements. Here's what the tabbing order of keyboard-focusable elements looks like for the top of the Accounts page.

The screenshot shows a form with the following elements and their tabbing order:

- 1: Nickname text input field (value: Open Text Corp HQ - North America Support)
- 2: Watch All Tickets checkbox (label: Watch All Tickets)
- 3: Primary Phone Number text input field (value: 55 11 99855-6473)
- 4: Alternate Phone Number text input field
- 5: Fax text input field (value: #)
- 6: General Support link (text: General Support tic)
- 7: Save button
- 8: Cancel button

Other visible elements include: Primary Contact dropdown (value: Steven Lloyd Udarbe), End User Code text input field (value: EU00142725), and Account Address section (Address 1: 9711 Washingtonian Blvd, City: Gaithersburg, Zip or Postal Code: 20878, Country: United States).

- The "Primary Contact" and "End User Code" text is ignored by a screen reader when tabbing through the form
- The "Watch All Tickets" checkbox doesn't have the a `<label>` element programmatically associated with it so when a screen reader user tabs from the "Nickname" field to the "Watch all Tickets" checkbox, the screen reader only announces

Checkbox not checked

Screen reader users would not be able to find the "Watch" feature the way it is coded now nor would they hear the help text "Notify me of all ticket updates for this account by email."

```
<p>
<label for="watchAllTickets" class="watchAllLabel">
<input id="watchAllTickets1" name="watchAllTickets" type="checkbox">
```

```
value="true">

```

- A user tabbing through the form is sent from the "Fax" field to a link "General support ticket" without any context because of the way the "Account Address" box is forced into the `<form>` element. This should be a separate aside that does not interfere with filling out the form.

Conclusion

The My Support Portal is old code based on an inaccessible version of the Sencha Ext JS framework. There is only so much reworking and fixing that can be done with CSS tweaks and HTML hacks. This web application would ideally be rebuilt using accessible methods and coding best practices from the beginning. Like security and privacy concerns, accessibility is not going away.

Accessibility needs to be part of the process from start to finish, including:

- Business requirements
- Design requirements
- QA requirements and test cases
- Training of new employees
- Ongoing training and professional development for current employees
- Software to detect accessibility flaws

Additionally, many engineers are not familiar with **semantic HTML markup** and do not understand what it means for HTML to be semantic, which is foundational for accessibility. Web browsers make allowances for a lot of bad coding practices and do not force engineers to fix accessibility errors. It's only with proper HTML training along with web accessibility training that we will be able to produce and maintain an accessible web application internally.

Further Learning

The OpenText User Experience Design team has resources on accessibility in its design system guide including:

- [Accessibility info](#) – OpenText's commitment to accessibility
- [Design for all](#) – Learn about the many ways people interact with digital content
- [WCAG 2.0](#) – See which guidelines require conformance at OpenText
- [Laws and regulations](#) – OpenText's legal responsibilities

External resources

- W3C Web Accessibility Initiative: [Making the Web Accessible](#)

- edX (free 4-week course): [Introduction to Web Accessibility](#)
- Google Developers: [Fundamentals of Web Accessibility](#)
 - Udacity: [Web Accessibility by Google](#)

Other Sites

These are other websites and web applications identified as being customer-facing which require remediation to comply with the AODA:

- My Support Knowledge Base
- Developer.opentext.com
- Forums.opentext.com
- ePay.opentext.com
- Training Registry (*reviewed, partial remediation*)
- OT Connect (*under review, partial remediation*)
- Hosted documentation
- Covisint portal
- WWW.opentext.com (*reviewed, partial remediation*)
 - Campaigns
 - Enterprise World
 - Blogs.opentext.com
 - SearchOpenText.com (*reviewed*)
 - BusinessNetwork.opentext.com (*reviewed*)
 - Investors.opentext.com
 - Resources.opentext.com, e.g. <https://resources.opentext.com/cloud-summit-migrating-workloads-wb>

References

- Google Analytics. (2020, January 24). *Analytics*. Retrieved from https://analytics.google.com/analytics/web/?hl=en#/report/content-overview/a34876862w62564200p66764574/_u.date00=20190101&_u.date01=20190331/
- Government of Ontario, Canada. (2020, January 24). *How to make websites accessible*. Retrieved from <https://www.ontario.ca/page/how-make-websites-accessible>
- WCAG 2.0 Evaluation Methodology Task Force. (2014, July 10). *Website Accessibility Conformance Evaluation Methodology (WCAG-EM) 1.0*. Retrieved from <http://www.w3.org/TR/WCAG-EM/>
- World Wide Web Consortium. (2008, December 11). *Web Content Accessibility Guidelines (WCAG) 2.0*. Retrieved from <http://www.w3.org/TR/2008/REC-WCAG20-20081211/>