MOOC Program
General enquiries +61 7 3138 0754
www.qut.edu.au/study/open-online-learning

MOOC Accessibility Guide

Version: V1
Prepared by: QUT MOOC Team
Created: 1 June 2015

Contents

Version history

Version number	Date	Reason/comments/approvals
V1.0	10/06/2015	First draft completed and shared with TB. [JS]
V1.1	27/07/2015	Revisions and further completion based on feedback and further research [JS]
V1.2	28/07/2015	Fleshed out the ALT and ARIA sections with more detail, rearranged sections and added a new Hyperlink section based on feedback from Alex [JS]

Introduction

In accordance with Australian law and best web practices, we are continually striving to have all our content meet the WCAG 2.0 Level AAA accessibility guidelines. At a bare minimum we must meet the mandatory Level AA. In November 2009, the Online and Communications Council (OCC) endorsed WCAG 2.0, requiring all Australian, state and territory government websites to conform to the guidelines to meet WCAG 2.0 Level A by December 2012. The Secretaries' ICT Governance Board (SIGB) extended the requirement for Australian Government (Financial Management and Accountability Act 1997) (FMA Act) agencies to conform to WCAG 2.0 Level AA standard by December 2014. No conformance date has been set for Level AAA.

Taking into consideration that this is sometimes a laborious process that even the University as a whole is not fully meeting, consider this a rolling list of content review and actions—prioritised by most affected and budgetary considerations. Apart from the obvious benefit to impaired users and its trickle down effect of improving our course content for all users, tangible penalties through litigation are a real world concern.

Another consideration is where the Learning Management Systems (LMSs) we host our content on fail to meet accessibility guidelines and their developing efforts to meet them. For instance, edX (and by extension Open edX) has recently been through litigation and settlement on completely overhauling their LMS by 2018. For the most part, the platform meeting accessibility requirements is the responsibility of the vendor (EdCast/Open edX, Desire to Learn, Future Learn, etc.) However, every effort should be made to keep up to date with their improvements, and if there are any workarounds we can pursue for shortcomings in their systems.

The general gist of accessibility concerns has not changed much over time. It poses the overarching question:

Is the content equally available with no loss of context or functionality to end users with the most common visual or aural impairments?

Accessibility requirement references

- Australian Government Web Guide-Accessibility: the official government mandate for meeting WCAG 2.0
- Web Content Accessibility Guidelines (WCAG) 2.0 W3C Recommendation: the main overview and starting point for all the guidelines outlined here and beyond.
- Techniques for WCAG 2.0: the official support document for the WCAG 2.0 recommendations detailing tips and success and failures of various methods.

General content areas

The following is an overview of content areas and the current solutions we are using to meet their accessibility requirements. It is an attempt to distil the massive amount of detail that can be found in the WCAG but not to be construed as a substitute. The collection of materials on the W3C website are living documents that are constantly being improved as technologies change and better methods standardised. We should always refer back there for major decisions or to elevate any conformance to Level AAA (this document focuses mainly on meeting the current mandatory Level AA).

Visual design

Stylesheets are the main source for controlling visual design choices to meet accessibility. Where we have control over this is sometimes limited by the platform. It is often worthwhile to review a vendor's product and suggest any corrections to unmet accessibility requirements. It is also important to apply these same principles to any pre-produced course content (images, diagrams, videos, etc.). The main areas to consider are:

Text alternatives (Level A)

All non-text content should have a text alternative so it can be changed into other forms people need such as: large print, braille, speech, symbols or simpler language. This includes:

- **Controls, Input**: have names that describe their purpose (this is usually the domain of the CMS/LMS)
- **TIme-Based Media**: see *Audio & Video Subtitles* below.
- **Test**: assessment items can be tricky as at least a descriptive identification of non-text content is needed but that doesn't reveal the answer.
- **Sensory**: content created to create a sensory experience (visual art, sax solo, etc.) needs at least a descriptive identification. Sensory characteristics that such as shape, size, visual location and direction, or sound that are required to understand content should be minimised or have text alternatives. Text should also be able to be resized up to 200% without loss of functionality (Level AA).
- **Decoration**: non-text content that is purely decorative or used only for visual formatting can be implemented in a way that is ignored by assistive technology.

Semantic structure (Level A)

Content should be created such that it can be presented in different ways (simpler layout, responsive to any display size and all the formats referred to in the previous section) without losing context or structure.

The sequence of information, content structure and relationships between connected or disconnected content should be either clear within the text or programmatically determined.

Colour & Contrast (Level A-AAA)

Less-than optimal choices of colour and contrast between elements can make content inaccessible for us all. This becomes subtler and less obvious with increased forms of colour blindness in which some of the tools referred to at the end of this document are best used to confirm conformance with the standards. In summary for content to meet the following Levels they must have:

- **Level A**: use of colour can not be the only differentiation between information. It must also be accompanied by either: programmatic access to colour,
- **Level AA**: Visual presentation of text and images of text must have a contrast ratio of 4.5:1 (foreground/background). Exceptions are:
- Large-scale text (contrast ratio of 3:1)
- Incidental or decorative text
- Brand name logotypes.
- **Level AAA**: Visual presentation of text and images of text must have a contrast ratio of 7:1. Same exceptions as Level AA.

Keyboard navigation (Level A)

All content is required to be navigable exclusively by a keyboard. Much of meeting this requirement is on the onus of the LMS as this is where most actionable navigation elements (buttons, links, entry fields, etc.) are the domain of. However, in our own content, proper semantic layout and intended logical order must be set for any:

- buttons
- links
- embedded form fields

Tab order

A key element of navigating by keyboard is the ability to tab between controls.

Keyboard traps (where focus cannot be moved away from a component by keyboard alone) are not permitted.

Hyperlinks

Hyperlinks to internal or external URLs should always include a TITLE attribute for general usability. These will also be read by most screen readers but not all. To ensure meeting this requirement, it is generally agreed to include an ALT attribute as well. A screen reader will take this as precedent over the TITLE and not read it as well. The text for this can mirror exactly what is in the TITLE attribute except when:

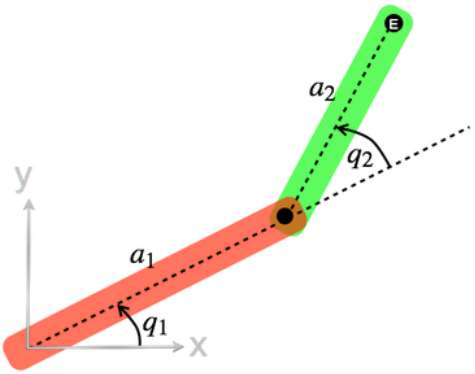
- further explanation to non-visual screen readers is deemed necessary
- an image is actually being linked (see Images section for further detail).

Images

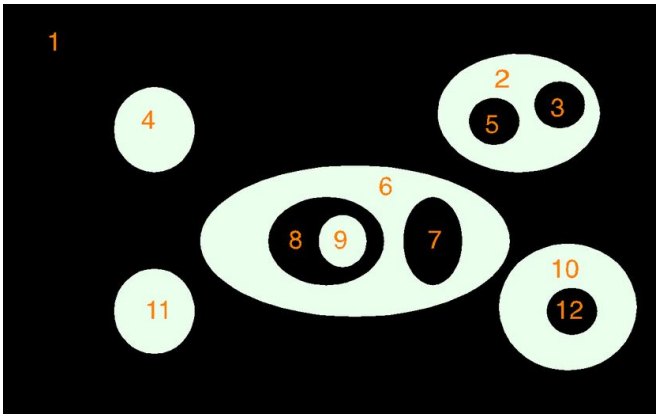
ALT attributes (only recognised by screen readers) are usually all that are required to meet accessibility requirements for images. As a starting point these may often replicate what is included in a TITLE attribute (which are visually displayed as tooltips) with mouse hover. Generally, it is necessary to go further than this with the non-visual description however.

A good example of a successful ALT text attribute will not simply give the title of the file, but describe what the image portrays, taking into consideration what is most relevant in the context of the learning. This can further be complicated by images that are part of an assessment item and caution must be taken to describe it as a non-impaired viewer would see without giving away the answer. Often a SME (Subject Matter Expert) is best to determine this but most general ones can be written sufficiently by a learning designer or technical team member.

Some good examples of such ALT attributes include:



ALT text: diagram of a two-link arm.



ALT text: blob 1 containing 5 blobs: 3 of which contain 1, 2, and 3 blobs respectively.

ARIA (Accessible Rich Internet Applications) properties

Semantic information added to content and other web elements allows assistive technologies to convey the context of roles, states, and properties that would otherwise go unrecognised by persons with disabilities. The W3C draft goes into great detail on the plethora of ARIA information that can be added, and for developers their Github index details much of this implementation. The following is what we prioritise from these 'present only to screen reader' attributes as our course content has required.

aria-label

An aria-label can be used to describe the *accessible name* for all other content besides images. This is worth considering and easy to implement especially when:

- the understanding or context is predominantly conveyed using visual accessories such as: groupings of paragraphs or other sections.
- groups of images that make more sense when described collectively, in addition to individually.
- LaTeX and similar syntax that is interpreted visually by a browser but fails to convey the intended meaning in its raw code.

W3C Examples

NOTE: **aria-labelledby** should be alternatively used when the label text is visible on screen.

aria-level

The aria-level can be used to define the hierarchal level of elements within the page structure.For instance when:

- the hierarchy or other organisation of said content is made further evident by visual accessories such as: prominence, tabbed or accordion sections, and other arrangements on a page.

Audio & Video Subtitles (Time-based Media)

Captions (Level AA)

Captions are text transcripts of both the speech and non-speech audio information needed to understand the media content. This aids *aurally-impaired* people that may have issues with the material at varying degrees.

Video Descriptions (Level AA)

Video descriptions (also audio descriptions of pre-recorded video or descriptive narrative) are written transcripts of the visible action that *vision-impaired* people may have issues with at varying degrees. This includes information about actions, characters, scene changes, on-screen text, and other visual content that is not otherwise described in the audio. To date we have not been capturing this in our video transcription process.

PDFs

Documents must meet a number of accessibility guidelines mainly around conveying the logical order of content independent of its appearance or layout. This is achieved by generating a Tagged PDF file. Generally, the PDF file format is not acceptable; however, in instances of very simple text documents, specific caveats in the guidelines allow them through. See PDF Techniques for WCAG 2.0 for full details.

> More complex documents might be better served as Word documents; however, they would also need to adhere to the same accessibility guidelines for PDF documents. The use of built-in styles will convey logical order. Image ALT tags and proper URL links with display text are all possible.

We use PDFs in our course content for text-only transcripts of videos. Other acceptable PDFs might include

supplementary reference material to a course.

It is possible to create accessible PDFs using the Adobe Acrobat PDF export plugin from your word processor if it is installed in your system. Saving as or Printing to PDF otherwise will not result in an accessible PDF. Regardless of the method used, all PDFs should be run for a final check through **Adobe Acrobat Pro** where it will likely need:

- the language specified in File Properties
- Tagging structure added or corrected.

Tables

Tables should be limited to *tabular data* only. Using tables as a layout aid to convey text broken down into sections is outmoded. The preferred method in lieu of *layout tables* is using style sheets or simply making a hierarchal structure through HTML heading levels or displaying in more interesting ways such as vertical accordions and horizontal tabs.

Tabular data

Things to ensure that are included are:

- **table header**: each column must include a table header cell that describes the content of that column.
- **headers** (optional): in more complex tables where groups of data cells are associated with each column must include a table header cell that describes the content of that column.
- **axis** (optional): this is only recommended if a complex table has sets of rows or columns contextually grouped. the benefit to accessibility is still up for debate.
- **captions** (optional): Include if the content of the table is not immediately obvious. Captions are also visibly display above/below the table to all visual users as well.
- **summary** (optional): Using a caption is preferred first. You would only additionally include a summary if an extended definition of the table seems necessary to convey the meaning that may be more obvious visually. A summary is not visible and only read by screen readers.
- if the tabular data is not clearly described by using the above, there are numerous other optional elements that can be included which are detailed in WCAG 2.0 HTML Techniques for Data Table.

The content of the table will determine which of the above elements are best needed and the development team can ascertain this when building if everything is clearly described.

Here is an example that illustrates the potential usage of these elements and how they are interpreted Visually, Aurally, and the code that generates this.

Visual display:

Daily MOOC coffee consumption

	Name	Cups	Type of Coffee	Sugar?
Morning				
7:30	Jason	2	Cappuccino	No
8:30	Andrew	3	Latte	No
8:15	Tabetha	1	Cappuccino	Yes
9:30	Liz	1	Flat white	No
9:45	Luisa	1	Mocha	No
10:00	Luzette	1	Green tea	No
Afternoon				
2:30	Jason	2	Cappuccino	Yes
2:30	Andrew	4	Latte	No

Aural screen reading:
Caption: Daily MOOC coffee consumption Summary: This table charts the number of cups of coffee consumed by the MOOC team split across mornings and afternoons. It specifies each team member, the type of coffee, and whether taken with sugar.
Morning 7:30 Name: Jason, Cups: 2, Type: Cappuccino, Sugar: No 8:30 Name: Andrew, Cups: 3, Type: Latte, Sugar: No… and so on.

Code:

```
<table border="1" summary="This table charts the number of
 cups of coffee consumed by the MOOC team split across mornings and afternoons. It specifies each team
member, the type of coffee, and whether taken with sugar.">
 <caption>Daily MOOC coffee consumption</caption>
 <tr>
  <th></th>
  <th id="name">Name</th>
  <th id="cups">Cups</th>
  <th id="type" abbr="Type">Type of Coffee</th>
  <th id="sugar">Sugar?</th>
 </tr>
 <tr>
  <th id="morning" axis="morning">Morning</th>
 </tr>
 <tr>
  <th headers="morning">7:30</th>
  <td headers="name">Jason</td>
  <td headers="cups">2</td>
  <td headers="type">Cappuccino</td>
  <td headers="sugar">No</td>
 </tr>
 <tr>
  <th headers="morning">8:30</th>
  <td headers="name">Andrew</td>
  <td headers="cups">3</td>
  <td headers="type">Latte</td>
  <td headers="sugar">No</td>
 </tr>
 <tr>
  <th headers="morning">8:15</th>
  <td headers="name">Tabetha</td>
  <td headers="cups">1</td>
  <td headers="type">Cappuccino</td>
  <td headers="sugar">Yes</td>
 </tr>
 <tr>
  <th headers="morning">9:30</th>
  <td headers="name">Liz</td>
  <td headers="cups">1</td>
  <td headers="type">Flat white</td>
  <td headers="sugar">No</td>
 </tr>
 <tr>
  <th headers="morning">9:45</th>
  <td headers="name">Luisa</td>
  <td headers="cups">1</td>
  <td headers="type">Mocha</td>
  <td headers="sugar">No</td>
```

```
  </tr>
 <tr>
    <th headers="morning">10:00</th>
    <td headers="name">Luzette</td>
    <td headers="cups">1</td>
    <td headers="type">Green tea</td>
    <td headers="sugar">No</td>
  </tr>


 <tr>
    <th id="arvo" axis="arvo">Afternoon</th>
  </tr>
  <tr>
    <th headers="arvo">2:30</th>
    <td headers="name">Jason</td>
    <td headers="cups">2</td>
    <td headers="type">Cappuccino</td>
    <td headers="sugar">Yes</td>
  </tr>
  <tr>
    <th headers="arvo">2:30</th>
    <td headers="name">Andrew</td>
    <td headers="cups">4</td>
    <td headers="type">Latte</td>
    <td headers="sugar">No</td>
  </tr>
</table>
```

Specialised Content Areas

LaTeX equations

This would be specific to our Engineering courses such as Robotics. The display of mathematical equations using the LaTeX markup language is accomplished by programmatic code that generates an image. This is because the mathematical symbols and correct layout is not possible with any character set alone, thus making it impossible to display or ALT tag it by text alone. There are two suggested methods currently in the WCAG 2.0 Techniques for LaTeX equations:

1. Provide the actual raw LaTeX code as a text download accompanying the image. A user can then download it and use with a dedicated interpreter such as AsTeR.
2. If LaTeX equation is generated as an image, provide a single description of the equation in an ALT tag or wrapping the content in an aria-label description if the equation is broken up between several images or read as LaTeX syntax code.

QA accessibility review

There are four phases of QA checking that may be utilised depending on content & resource factors:

1. Tools that perform standardised checks such as **Screen readers** and **Colour contrast analysers**.
2. Review of specialised areas based on agreed solutions (LaTeX for example).
3. Review of subjective solutions. A Screen reader can confirm ALT tags or aria-labels are present, but not if it satisfactorily meets the requirements and context (instruction, assessment). A SME should not be necessary for this phase, provided they were consulted during the content creation.
4. [Optional] Usability testing by actual impaired users. This should not be required to clear the content

except in extreme cases of uncertainty or where resources provide.

A number of tools are available to run accessibility checks on content. Most notable are **Screen readers,** which are the actual browsers or browser plugins that vision-impaired visitors would use to navigate a site.

Site validation

- WebAIM WAVE Accessibility Checker: a comprehensive check of a given URL on all accessibility concerns.

Screen readers

The are a number of screen readers available on the market from Free to Paid. The most thorough one on Windows is JAWS (Commercial). WebAIM has a quick guide on using it for accessibility evaluation purposes. Free options include: NVDA (Win), Orca (Linux), ChromeVox (Chrome) and WebAnywhere (Web). On Mac/iOS the free built-in Voiceover can easily be turned on in System Preferences. Further advantages to Voiceover are that it's system-wide (meaning it seamlessly assists accessibility throughout the computer), supports multi-touch gestures and has plug and play support for refreshable braille displays. Again WebAIM has a quick guide on use for evaluation purposes.

We use these at the meta level to test if developing style practices are meeting accessibility requirements for the layout and labelling of elements.

During the QA process this should be incorporated as part of the procedures for checking that all content is laid out and labelled correctly. This process entails:

- loading up the chosen Screen Reader browser
- navigating to the site
- clicking through all the content and aurally confirming that:
    - the navigation is in the logical order and not jumping around the page
    - sections and media have titles and alt tags present where needed and not generically unlabelled.

Colour contrast tools

In the design phase, Photoshop provides several built-in tools for colour-proofing. There are a number of free tools on the web that also simulate varying colour-blindness such as Paletton. The best source for validating Foreground/Background colour choices is the WebAIM Color Contrast Checker.

While viewing content already in place in a platform or during the QA process, several website services can analyse the color palettes used on a provided URL. These include:

- Colors on the Web
- Colorfyit

Specialised checklists

In addition to standardised tests we would also need to incorporate a checklist of some of our specialty areas and their agreed upon solutions to confirm they have been implemented. In our experience so far this would include field-specific material such as:

- LaTeX mathematical syntax or similar markup that is non-text and non-image based.

Usability testing

Testing by actual impaired users could be incorporated within a larger phase of usability testing for content correctness and User Experience (UX) evaluation. If all other phases of review are completed during production this shouldn't be mandatory except in extreme cases where their expertise would be needed. Resources or personnel may sometimes be available regardless in which case this would obviously be the most thorough final check.