Mobile Accessibility Testing

Methodology

Step 1: Identify what needs to be tested
Step 2: Conduct specific mobile tests
Mobile Testing Methodology Overview

Please note that this methodology does not include those errors already included in WCAG2. In order to ensure your mobile site is fully accessible, you need to meet WCAG2 and this mobile testing methodology.

**Step 1: Identify what needs to be tested**

**Identify devices**

Recommended devices:
- iPhone, Safari
- iPad, Safari
- Android phone, Chrome
- Android tablet, Chrome

**Identify the site type and variations of the page**

There are three types of sites on the web and each type has different mobile testing requirements:
- **Desktop web sites**: that have only one display, whether viewed on desktop or mobile or tablet device;
- **Responsive web sites**: that change depending on the screen size or other feature as determined by the developer;
- **m.dot sites**: that have a particular display for mobile and tablet sites. The m.dot site must also be tested against the entirety of WCAG2, in addition to the standard www version of the site.

Responsive sites contain multiple variations or versions of a page. It is important that each variation of the page is tested and that all functionality is available on all variations of the page. People with disabilities may be restricted to one variation of the page only. Developers can vary one or both of the following:
- Variation in content included on the page; and
- Variation in the presentation of components displayed.

**Step 2: Conduct specific mobile tests**

In addition to the familiar criteria that are tested on desktop – such as alternatives to images, and coding headings and tables – there are five more types of mobile testing errors:

a. **Critical mobile-specific interoperability**: hover trap, touch trap, screen reader swipe trap, on-screen keyboard trap, zoom trap, etc.

b. **Mobile-specific interaction**: orientation, motion actuation, geolocation, scroll-bars, blocking pinch zoom, touch targets, inactive space, navigational aids, etc.

c. **Mobile assistive technology support**: screen reader support, keyboard behavior, magnification / zoom behavior, inverse colors / grayscale behaviour, etc.

d. **Mobile and desktop relationship errors**: consistency, restriction of content, choice of content, etc.

e. **Non-specific mobile issues common on mobile**: alternatives for items only displayed in mobile (eg. hamburger menus), underlined links, reference to attributes, etc.
# Table of Contents

Mobile Testing Methodology Overview ................................................................. 2
Step 1: Identify what needs to be tested ........................................................................ 2
Step 2: Conduct specific mobile tests ............................................................................... 2

Table of Contents ........................................................................................................ 3

Introduction .................................................................................................................. 4
A note on mobile apps ........................................................................................................ 4

About mobile testing ........................................................................................................ 5
Types of mobile sites ........................................................................................................ 5
Testing methods ................................................................................................................ 8
Capturing errors .............................................................................................................. 12
Examples of mobile accessibility errors ............................................................................. 14
Variations of the page (responsive sites only) ................................................................. 17

Mobile Testing Methodology .......................................................................................... 20
Identify what needs to be tested ...................................................................................... 20
Test critical mobile-specific interoperability ................................................................. 22
Test mobile-specific interaction ...................................................................................... 22
Test mobile assistive technology support ....................................................................... 23
Test mobile and desktop relationship errors ................................................................. 24
Test non-specific mobile issues common on mobile ....................................................... 24

Acknowledgements ......................................................................................................... 26

Relationship to existing Accessibility testing standards ................................................. 26
ICT Accessibility Testing Symposium Mobile Sub-Committee ......................................... 26

Contacts ...................................................................................................................... 26
Gian Wild .................................................................................................................... 26
Introduction

ICT Accessibility Testing Symposium has developed a methodology for evaluating the accessibility of mobile web sites. This document is an amalgamation of accepted mobile accessibility testing standards from around the world, including additional developments from the ICT Accessibility Testing Symposium’s Mobile Sub-Committee (for more information, see Acknowledgements on page 25).

WCAG2 success criteria are applicable to mobile, however, not all aspects of mobile accessibility are specifically covered by WCAG2. For example, although WCAG2 requires sites to be accessible to the keyboard user, it does not specify that it should also be accessible to the touchscreen user. WCAG2.1 builds on this and addresses more criteria related to touch screen (pointer gestures), sensors and small screen devices, however it still does not cover all user needs related to mobile accessibility.

Please note that this methodology does not include those errors already included in WCAG2. In order to ensure your mobile site is fully accessible, you need to meet WCAG2 and this mobile testing methodology.

A note on mobile apps

There is a great difference between mobile sites and mobile apps – native apps utilize a completely different codebase. Therefore, the ICT Accessibility Testing Symposium has decided to focus on a mobile site testing methodology only. It is expected that a mobile app testing methodology will be developed in the future.
**About mobile testing**

Please note that this methodology does not include those errors already included in WCAG2. In order to ensure your mobile site is fully accessible, you need to meet WCAG2 and this mobile testing methodology.

**Types of mobile sites**

There are three types of sites on the web:

- **Desktop web sites**: that have only one display, whether viewed on desktop or mobile or tablet device;
- **Responsive web sites**: that change depending on the screen size or other feature as determined by the developer;
- **m.dot sites**: that have a particular display for mobile and tablet sites. The m.dot site must also be tested against the entirety of WCAG2, in addition to the standard www version of the site.

Most sites are responsive sites. All these site types need to be tested on mobile devices, however testing is different depending on the site type:

- Mobile and Desktop relationship errors **do not** need to be tested on Desktop web sites.
- Some Mobile and Desktop relationship errors do not need to be tested on Responsive web sites.
- In addition to this mobile testing methodology, **all WCAG2 errors** need to also be tested on m.dot sites.

**Desktop web site**

The site is a desktop site if the content does not change as you drag the browser window narrower. In most cases you will see a horizontal scrollbar at the bottom of the page.

**Example of a desktop web site**

*Figure 1: Australian National Botanic Gardens desktop web site at desktop size*

*Figure 2: Australian National Botanic Gardens desktop web site at mobile window size*
Responsive web site

Please note that it is most likely that a site is responsive. If a site is responsive, it is definitely not a desktop site. However, it is possible (but very unlikely) that there is also a mobile web site.

The site is responsive if the layout changes as you change the browser window size. To test this, open the web site in a browser, ensure the window is not maximized, and select the right-hand edge and drag it to the center of the screen. If elements in the page move around, then the site is responsive. Another way to tell is if the navigation disappears and is replaced with a hamburger menu.

Example of a responsive web site

Figure 1: AccessibilityOz site at desktop width

Figure 2: AccessibilityOz site at a mobile width
m.dot site

In most cases you will need to ask the client if they have an m.dot site. However, most mobile sites are the same as the desktop URL but preceded with “m.” instead of “www.”. Therefore, you can type in the URL preceded by “m.” to determine if the site has a mobile version. However, please make sure that the site does not automatically redirect to the “www.” site – this can be checked by looking at the URL address bar once the site has loaded. Another way to tell is to open the “m.” site in a desktop-sized window and it will still look like a mobile site. You can then check how the site changes when you change the size of the browser window.

In some cases, a site will only display the mobile site if opened in a mobile device. To determine if this is the case, open the site on the desktop and reduce the window size so it is the same width as a mobile browser. Open the same site on your mobile phone and determine if the site is different. If the site is different then there is a separate mobile site.

Comparison of a desktop and m dot site

Figure 3. www.sephora.com – their desktop site in a desktop-sized window
Testing methods

There are five main testing methods in mobile testing:

- **Device**: test on mobile and tablet devices
- **Device with assistive technology**: test on mobile and tablet devices with assistive technologies
- **Simulator**: test on mobile and tablet simulators
- **Window**: test on different sized window on desktop
- **Desktop**: test on desktop
Devices

Due to the popularity of the Android system, there are tens of thousands of Android operating systems and browser combinations available. It is not possible to test on all these systems.

Please note that the “Internet” browser that comes pre-packaged with most Samsung phones is very dependent on the operating system itself and it is a better representation to test with Chrome.

Note: an assumption is that mobile sites will support touch access because that is the dominate mode of use by typical users.

Recommended devices and browser combinations

- iPhone, Safari
- iPad, Safari
- Android phone, Chrome
- Android tablet, Chrome

Determining which devices to test on

In the United States and other western countries, iOS devices are most popular. In Asia and other eastern countries, Android devices are most popular. To best find which devices to test on, review the Google Analytics or other analytics system for the requisite web site.

Device with assistive technologies

Even if the site is a desktop site, people will still use the site on mobile with various assistive technologies. In some cases, as with VoiceOver, these assistive technologies are also available on desktop, however in most cases, assistive technologies are inherent to the device. It is important to remember that even assistive technologies that work across desktop and devices may behave differently on each system, and therefore they still need to be tested on mobile and tablet devices.

For the latest information on screen reader usage, please see the WebAIM Screen Reader Survey.

Samsung includes an additional screen reader called “Voice Assistant,” however TalkBack is still available as part of the Accessibility Suite. Amazon Fire also utilizes a different screen reader called “Voice View”. These may need to be incorporated into testing.

The following assistive technology / device combinations should be tested:

- VoiceOver on iOS
- TalkBack on Android
- Keyboard on iOS
- Keyboard and switch on Android
- Zoom on iOS
- Magnification on Android

The following only need to be tested if there is a variation in content. For more information see Variations of the page (responsive sites only) on page 17.

- Invert colors on iOS
- Grayscale on iOS
- Grayscale on Android
Using VoiceOver on iOS devices

VoiceOver users include blind and low-vision users, as well as users with some cognitive or learning difficulties. To enable VoiceOver, navigate to Settings → General → Accessibility → VoiceOver. Then, touch and slide the white Off button (beside the word VoiceOver) to the right to enable VoiceOver. Once in the On position, the button changes color to indicate VoiceOver is enabled.

Using TalkBack on Android devices

To enable TalkBack, navigate to Settings → Accessibility → TalkBack then switch the button to On. A “Use TalkBack” confirmation screen will appear. When used for the first time a tutorial will be launched. Ensure the device volume is turned up as TalkBack doesn’t automatically adjust the volume if muted.

Using keyboard on iOS devices

Connect a keyboard to your iOS device. Use as you would a keyboard on the desktop.

Using keyboard on Android devices and Switch Access

To enable switch access, navigate to Settings → Accessibility → Switch access and then switch the button to On. Switch access allows you to control your device using configurable key combinations. This means you can use an external keyboard or switch device to execute core keyboard commands to navigate and interact with web sites and native apps. As with the use of TalkBack, this is a useful feature for testing keyboard access and content order.

To use switch access, you’ll need a keyboard paired with the tablet or mobile. Then you’ll need to assign keys to actions through Settings → Accessibility → Switch access → Settings. Select Actions under the heading “Assign Keys to Actions” to map a keystroke to the action.

Using Zoom on iOS devices

People with low vision often use Zoom. Users with cognitive impairments, such as autism, may also use Zoom to focus in on key content and reduce the amount of visual clutter on the screen. To enable full screen Zoom, first navigate to the Zoom screen and select Settings → General → Accessibility → Zoom.

Then, touch and slide the white Off button (beside the word Zoom) to the right to enable Zoom. Once in the On position, the button changes color to indicate Zoom is enabled.

Standard iOS gestures—flick, pinch, tap and rotor—still work when the screen is magnified. Invert Colors and Grayscale also work with Zoom. There are some differences in commands when Zoom is run with VoiceOver.

You can view magnified screens either in either “Full Screen Zoom” or “Window Zoom.”

Using Magnification on Android devices

To enable Zoom, navigate to Settings → Accessibility → Magnification Gestures then switch the button to On.

When magnification gestures are enabled, you can zoom, pan across the screen and pan content:

- Zoom in or out by triple-tapping the screen. Then, while zoomed in, you can:
  - Drag two or more fingers to pan across the screen.
Pinch two or more fingers together or spread them apart to adjust the zoom level.

Using Invert colors with iOS

To enable Invert Colors, first navigate to Settings → General → Accessibility → Display Accommodations and find Invert Colors in the Vision list.

Then, touch and slide the white Off button (beside the words Invert Colors) to the right to enable Invert Colors. Once in the On position, the button changes color and display colors are inverted to indicate Invert Colors is enabled.

The color inversion is applied to all screens until the feature is disabled by sliding the button to the Off position.

Invert Colors can be used with VoiceOver, Zoom and Grayscale.

Using Grayscale view with iOS

To enable Grayscale, first navigate to Settings → General → Accessibility → Display Accommodations → Color Filters Accommodations and enable the Color Filters toggle to view greyscale and other filters.

Then, touch and slide the white Off button (beside the words Grayscale) to the right to enable Grayscale. Once in the On position, the button changes to dark gray and display colors are replaced by grayscale colors to indicate Grayscale is enabled.

Grayscale is applied to all screens until the feature is disabled by sliding the button to the Off position.

Grayscale can be used with VoiceOver, Zoom and Invert Colors.

Using Grayscale view with Android

To enable Android’s grayscale view, first enable Developer Mode on your device. (This is a one-time process for each device.)

- Go to Settings → About Tablet (which might also be labelled “About Phone” or “About Device”).
- Locate the Build number section and tap on the phrase “Build number” 7 times.
- Go back to the main Settings page and tap on “Developer Options.”
- Enable the Developer Options switch.

From then on, you can enable Android’s grayscale view (Lollipop 5.0 onwards) by going to Settings → Developer Options → Simulate color space and selecting “Monochromacy.” Once you’ve made that selection, display colors will be replaced by shades of gray.

Grayscale is applied to all screens until you turn off the feature, which you can do by going back to Settings → Developer Options → Simulate color space and selecting “Disabled.”

Android’s grayscale view can be used alongside any of Android’s other modes or applications, including TalkBack and Shade, among others.

Simulators

It is not recommended that simulators be used for all testing, as they do not present reliable, accurate results. However, simulators can be used safely when testing some mobile errors.

Some mobile simulators include:
• XCode for iOS
• Android Studio

Paid simulators (have their issues):
• Browserstack
• Saucelabs

Window

Some errors can be tested on desktop by resizing the browser window.

Chrome’s developer tools also offer “device simulation”, which can be used for window testing. Open the developer tools (View -> Developer -> Development Tools), then select the “Toggle Device Toolbar” option:

This will cause the Device Toolbar to appear in the main browser window. From here you can select a device size to apply, along with other useful options:

Desktop

Some errors can be tested on desktop, using standard accessibility methods.

Capturing errors

There are two main methods of capturing mobile errors:

• Capturing code of the page
• Capturing screenshots or video of the page on the device

Once these have been captured, you can use standard automated accessibility testing tools to test the code and screenshots.

Capturing code of the page

The mobile DOM can be captured via:

• Developer tools on Safari (iOS only)
• Android – Developer mode
• Browser extensions
Capturing screenshots or video of the page on the device

There are three main methods for capturing screenshots or video of the page on the device:

- Taking a screenshot with the device and emailing it
- Replicating the screen of the device on a desktop and taking screenshots via a desktop app
- Recording or photographing the screen of the device with another device

Taking a screenshot with the device and emailing it

- **iPhone X**: To capture a screenshot, press and release the side button and the volume up button at the same time.
- **iPhone 6, 7, 8**: To capture a screenshot, press and briefly hold both the power button and the home button at the same time.
- **iPad**: To capture a screenshot, press and hold the Sleep/Wake (on/off) button on the top of your iPad. Quickly click the Home button at the bottom of the screen.
- **Samsung Galaxy phone**: To capture a screenshot, press and hold the Power and Volume down buttons at the same time (for approximately 2 seconds).
- **Samsung Tab A**: To capture a screenshot, simultaneously press and hold the Power button (located on the top-left edge) and the Home button.
- **Google Pixel**: To capture a screenshot, simultaneously press and hold the Power then tap the Screenshot icon. Alternatively, press and hold the Power and Volume Down buttons.

Direct recording from the device (iOS and iPad with iOS 11 and later)

From iOS 11 you can [directly record video from the device](#). First go to **Settings → Control Center** and select "Customize Controls". Add "Screen recording" to controls' list. Now the "Screen recording" control will be available in Control Center. Select it to start and stop recording.

Replicating the screen of the device on a desktop and taking screenshots via a desktop app

- **Reflector** works on iOS and Android. Once your phone screen is reflected on PC, you can take screenshot any time you want. Choose where you would like to save the screenshots by clicking the third icon under the Reflector. Afterwards, just click the camera icon under the Reflector.
- **SideSync** works on Samsung devices. Open SideSync on your computer and phone to have them connected. On your computer's SideSync, click “Phone screen.” Tap “v” on your phone screen window and select Capture. The screenshot images are saved to the folder that you've set in the SideSync settings.
- **ChromeCast** works on Android. ChromeCast is a mobile app that can mirror the device on a desktop or other screen.

Recording or photographing the screen of the device with another device

- **Taking a photo with another phone**: take a photo and then email it
- **IPEVO**: sends live video from your mobile device to a computer which can be recorded.
Examples of mobile accessibility errors

Hover traps
Content must be able to be dismissed if activated on touch (often these are actionable items that are activated on mouse hover on a desktop)

Touch traps
User must always be able to scroll / swipe to move up and down the page
Screen reader swipe traps

VoiceOver users must always be able to activate an item on the current page or move back to the previous page.

Onscreen keyboard traps

Onscreen keyboard must be able to be dismissed.
Zoom traps

Do not replace the entirety of the page with a feature that over-rides standard mobile functions such as swiping and scrolling.

Only by hitting the small white area bordering the map on the top and right is it possible to move up and down the page – otherwise only the map moves.
Variations of the page (responsive sites only)

Why provide different variations of a page?

Responsive sites contain multiple variations or versions of a page. It is important that **each variation of the page is tested** and that **all functionality is available on all variations of the page**. There may be many reasons that a site owner wants to provide variations of one particular web page, for example:

- Highlighting particular content on mobile, such as phone details;
- Hiding particular content on mobile, such as image galleries; and
- Hiding functionality that does not work on mobile, such as a drag and drop feature.

Developers can vary one or both of the following:

- **Variation in content** included on the page; and
- **Variation in the presentation** of components displayed.

Developers can identify one or more of the following four features to determine which content to include or display on a page:

- The **device** (e.g. iPhone, desktop, Android, etc.);
- The **operating system** (e.g. Windows, iOS, OS, etc.);
- The **browser** (e.g. Safari, IE 11, Chrome, etc.); and
- The **screen size** (e.g. 280 by 720, 1920 by 1080, 320 by 480, etc.).

A note on variations due to screen size

Low vision users (who use the zoom function inherent in the browser) are often restricted to a mobile view of the site on their desktop. As part of WCAG2, zooming to 200% should already be included in regular testing (and therefore is not included in this methodology). It is essential that functionality is not removed due to a variation in the page.

For example, previously in YouTube (this has now been fixed), the upload and notifications buttons were visible at 100% screen size but not at 200% screen size. This would mean that people browsing at 200% screen size would not be able to upload a video or view their notifications.

![YouTube at 100% zoom](image)

![YouTube at 200% zoom](image)
Types of variations of a page

Variation in content included on the page

This is where the content is varied dependent on a particular feature (device, operating system, browser, screen size).

Figure 9: AccessibilityOz site at desktop width

AccessibilityOz

Figure 10: AccessibilityOz site at a mobile width

Variation in the presentation of components displayed

This is where the content does not change, but the presentation of that content does.

Figure 11: ICT Accessibility Testing Symposium site at desktop width

Figure 12: ICT Accessibility Testing Symposium site at mobile width
Which variations of a page need to be tested

Every variation of a page needs to be tested, if it contains a variation in content or variation in presentation. For more information see WCAG2.1: Conformance.

People with disabilities may be restricted to one variation of the page only and may not be able to access other variations due to restrictions based on their accessibility needs. For example, people can be dependent on one particular device, orientation (people with vision impairments relying on screen readers often lock the screen in portrait mode), screen size or use of zoom.

How to test variations of a page

Although all functionality must be available on all versions of a page, the functionality does not need to be displayed in the same way. For example, content can be hidden via an expandable menu (e.g. a hamburger menu), included on another page via a link (e.g. a map moved to a secondary page with a link provided) or the functionality could be replaced with a different feature (e.g. a drag and drop feature replaced with a form that provides the same functionality). For more information see the WAI Tutorial page, when you zoom in to 300% the left-hand navigation moves to the bottom of a page, and a link “Jump to the navigation” is added.
Mobile Testing Methodology

Please note that this methodology does not include those errors already included in WCAG2. In order to ensure your mobile site is fully accessible, you need to meet WCAG2 and this mobile testing methodology.

Identify what needs to be tested

What needs to be tested is dependent on the site type, variations of the page and chosen devices.

Identify devices

Recommended devices and browser combinations:

- iPhone, Safari
- iPad, Safari
- Android phone, Chrome
- Android tablet, Chrome

Identify the site type and variations of the page

- Desktop web sites: that have only one display, whether viewed on desktop or mobile or tablet device;
- Responsive web sites: that change depending on the screen size or other feature as determined by the developer;
- m.dot sites: that have a particular display for mobile and tablet sites. The m.dot site must also be tested against the entirety of WCAG2, in addition to the standard www version of the site.

The testing methods will be dependent on what site type is being tested, and, in the case of responsive sites, how variations in content are controlled.

Desktop web site testing

<table>
<thead>
<tr>
<th>Types of errors</th>
<th>Testing methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical mobile-specific interoperability</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile-specific interaction</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile assistive technology support</td>
<td>Device and assistive technology</td>
</tr>
<tr>
<td>Non-specific mobile issues common on mobile</td>
<td>Simulator or Window or Device</td>
</tr>
</tbody>
</table>

Responsive web site testing

It is important that each variation of the page is tested and that all functionality is available on all variations of the page. The testing methods for responsive web site testing are dependent on whether there variations of the page.

Determine if there is:

- Variation in content included on the page; and
- Variation in the presentation of components displayed.
Determine which of the following triggers the variation of the page:

- The **device** (e.g. iPhone, desktop, Android, etc.);
- The **operating system** (e.g. Windows, iOS, OS, etc.);
- The **browser** (e.g. Safari, IE 11, Chrome, etc.); and
- The **screen size** (e.g. 280 by 720, 1920 by 1080, 320 by 480, etc.).

<table>
<thead>
<tr>
<th>Types of errors</th>
<th>Variation in content via:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Device</td>
</tr>
<tr>
<td>Critical mobile-specific interoperability</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile-specific interaction</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile assistive technology support</td>
<td>Device and assistive technology</td>
</tr>
<tr>
<td>Mobile and desktop relationship errors</td>
<td>Device and Desktop</td>
</tr>
<tr>
<td>Non-specific mobile issues common on mobile</td>
<td>Simulator or Window or Device or Desktop</td>
</tr>
</tbody>
</table>

**m.dot site testing**

The m.dot site must also be tested against the entirety of WCAG2, **in addition** to the standard www version of the site.

<table>
<thead>
<tr>
<th>Types of errors</th>
<th>Testing methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical mobile-specific interoperability</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile-specific interaction</td>
<td>Device</td>
</tr>
<tr>
<td>Mobile assistive technology support</td>
<td>Device and assistive technology</td>
</tr>
<tr>
<td>Mobile and desktop relationship errors</td>
<td>Device and Desktop</td>
</tr>
<tr>
<td>Non-specific mobile issues common on mobile</td>
<td>Simulator or Device</td>
</tr>
</tbody>
</table>
Test critical mobile-specific interoperability

Hover traps
Content must be able to be dismissed if activated on touch (often these are actionable items that are activated on mouse hover on a desktop). For more information see WCAG2.1 SC 2.5.1: Pointer Gestures.

Touch traps
User must always be able to scroll / swipe to move up and down the page

Screen reader swipe traps
VoiceOver users must always be able to activate an item on the current page or move back to the previous page.

Onscreen keyboard traps
Onscreen keyboard must be able to be dismissed.

Zoom traps
Do not replace the entirety of the page with a feature that overrides standard mobile functions such as swiping and scrolling.

Test mobile-specific interaction

Orientation
The system can be used in portrait mode (for more information see WCAG2.1 SC 1.3.4: Orientation).
The system can be used in landscape mode (for more information see WCAG2.1 SC 1.3.4: Orientation).
Users can switch between portrait and landscape mode (for more information see WCAG2.1 SC 1.3.4: Orientation).
Horizontal scroll bars do not appear when switching from one orientation to another.

Motion Actuation
Functionality that can be operated by device motion or user motion can also be operated by user interface components, and responding to the motion can be disabled to prevent accidental actuation, except when:

- **Supported Interface**: The motion is used to operate functionality through an accessibility supported interface; or
- **Essential**: The motion is essential for the function and doing so would invalidate the activity.
For more information see WCAG2.1 SC 2.5.4: Motion Actuation.

Alternatives
Alternatives are provided for non-web mobile functionality that is mandatory (for example, recording a specific gesture by the camera, requiring a specific geolocation before functionality appears).
Important functionality and information is available in the Reader view
Keyboard

Do not present new content on hover over actionable element (for example, do not have a top-level menu item that displays sub-items on hover, but also when tapped opens a new page).

On-screen keyboards or other pop-up components can be dismissed.

Alerts use a dialog box that is standard to the mobile.

Touch targets

Range of touch targets is at least 7 to 10mm (i.e. no smaller than the smallest average finger).

Actionable elements are triggered only on removal of touch (ON TOUCH START and ON KEY DOWN have not been used) (for more information see WCAG2.1 SC 2.5.2: Pointer Cancellation).

Support for alternative input methods aside from touch are supported (for more information see WCAG2.1 SC 2.5.1: Pointer Gestures).

When direct input via the keyboard is not required provide options for the user to achieve the same result (i.e. use dropdown, radio buttons & checkboxes, etc.) (for more information see WCAG2.1 SC 2.5.1: Pointer Gestures).

Inactive space

Touch targets/actionable items have sufficient inactive space between them (Inactive space of at least 1 pixel should be provided around active elements – not currently identifiable programmatically).

Resize

Horizontal scroll bars do not appear at all when the page is resized.

Pinch zoom is operable (if users are unable to use the zoom feature then font resizing should be provided allowing the user to increase the size of content at least two times the size of the standard font size).

Navigational aids

Arrows and Next and Previous buttons have been used to indicate swipe or scroll areas (for more information see WCAG2.1 SC 2.5.1: Pointer Gestures).

Active swipe elements support both horizontal scrolling and swipe gestures.

Back buttons move the user back to the previous step (i.e. do not act like a breadcrumb trail).

Toggle and slider elements can be dragged and support touch gestures as a fallback (for more information see WCAG2.1 SC 2.5.1: Pointer Gestures).

Test mobile assistive technology support

ARIA-HIDDEN=TRUE has not been used to hide content essential to the screen reader user
DISPLAY: NONE has not been used to hide content essential to the screen reader user
ROLE=PRESENTATION has not been used to remove required functional relationships
Changes of state of non-standard controls (e.g. hamburger menu, star ratings) are clearly indicated
All actionable items can be accessed and activated by a device screen reader
All important content can be accessed by a device screen reader
All actionable items can be accessed and activated via a keyboard attached to a device
Device zoom / magnification can be implemented without horizontal scrolling or overlapping content
Invert colors / grayscale is inherited by all actionable items and important content

Test mobile and desktop relationship errors

Testing on desktop
Item labelling across mobile and main site is consistent
Links between mobile and full version of the web site have been provided
Users are not restricted to a particular version dependent on device (i.e. cannot use mobile version on desktop and vice versa)

Testing on device
Item labelling across mobile and main site is consistent
Links between mobile and full version of the web site have been provided
Users are not restricted to a particular version dependent on device (i.e. cannot use mobile version on desktop and vice versa)

Test non-specific mobile issues common on mobile

Actionable items
Standard UI controls, objects and elements have been used
Context changes only with warning on scrolling to the end of the page
Changes of state of non-standard controls (e.g. hamburger menu, star ratings) are clearly indicated
Audio cues also have a visual cue
JavaScript OnChange should be checked
ARIA document landmarks have been used to (correctly) describe document structure
ARIA tooltips have not been used alone

Audio and Video
On mobile, the lack of a transcript is acceptable IF captions have been provided.
Slightly different focus for mobile: audio must not play automatically unless the user is:
  • made aware this is happening; or
  • a pause/stop button is provided.

Links
Links are underlined
Link to a new web site is clear
Links to a single page or resource are not repeated unnecessarily on the same page
Image links do not have incorrect ALT attributes (“Link to...”)
Forms

Field labels are positioned above (portrait mode) or to the left (landscape mode) of the input field
Field labels for radio buttons or checkboxes are positioned to the right of the control
The HTML5 input type is correct
Radio buttons or checkboxes within a FIELDSET have the same NAME

Images

Background images have not been used to convey information

Navigation

Navigation text uniquely describes the target of the navigation item

Content

References to visual location or shape (e.g. above, below, box) alone have not been used to convey information
References to size alone (e.g. large, small) have not been used to convey information
References to attributes alone (e.g. bold, italicized) have not been used to convey information
Navigational aids such as back buttons, arrows, next and previous buttons are provided
Acknowledgements

Relationship to existing Accessibility testing standards

This document is based on:

- W3C Web Content Accessibility Guidelines, Version 2.0
- W3C Web Content Accessibility Guidelines, Version 2.1
- BBC Mobile Accessibility Guidelines
- AccessibilityOz Mobile Testing Methodology
- TPG Mobile Testing Guide

ICT Accessibility Testing Symposium Mobile Sub-Committee

This document was developed by the ICT Accessibility Testing Symposium Mobile Sub-Committee. Members include: Gian Wild (Co-Chair), Sunish Gupta (Co-Chair), Jonathan Avila, Jennifer Chadwick, Matt Feldman, Peter McNally, Alyson Muff, Andrew Nielson, Laura Renfro, Janet Sedgley and Chris Law.

Contacts

Gian Wild

Web:  www.accessibilityoz.com
Phone:  415 621 9366
Email:  gian@accessibilityoz.com