What is accessibility?

↑ Overview: Accessibility Next →

This article starts the module off with a good look at what accessibility actually is — this includes what groups of people we need to consider and why, what tools different people use to interact with the web, and how we can make accessibility part of our web development workflow.

Prerequisites:	Basic computer literacy, a basic understanding of HTML and CSS.
Objective:	To gain familiarity with accessibility including what it is, and how it affects you as a web developer.

So what is accessibility?

Accessibility is the practice of making your websites usable by as many people as possible — we traditionally think of this as being about people with disabilities, but really it also covers other groups such as those using mobile devices, or those with slow network connections.

You could also think of accessibility as treating everyone the same, and giving them the same opportunities, no matter what their ability or circumstances. In the same way that it is not right to exclude someone from a physical building because they are in a wheelchair (public buildings generally have wheelchair ramps or elevators these days), it is also not right to exclude someone from a website because they have a visual impairment, or are using a mobile phone. We are all different, but we are all human, and therefore have the same (human) rights.

Accessibility is the right thing to do, but it is also part of the law in some countries, and it can open up some significant markets that otherwise wouldn't be able to use your services, buy your products, etc.

Accessibility and the best practices it entails can benefit everyone:

- Semantic HTML (which improves accessibility) also improves SEO, making your site more findable/marketable.
- Caring about accessibility demonstrates good ethics/morals, which improves your public image.
- Other good practices that improve accessibility also make your site more usable by other groups, such as mobile phone users, those on a low network speed, etc. In fact, everyone can benefit from many such improvements.
- Did we mention it is also the law in some places?

What kinds of disability are we looking at?

People with disabilities are just as diverse as people without disabilities, and so are their disabilities. The key lesson here is to think beyond your own computer and how you use the web, and start learning about how others use it — *you are not your users*. The main types of disability to consider are explained below, along with any specialist tools they use to access web content (known as **assistive technologies**, or **ATs**).

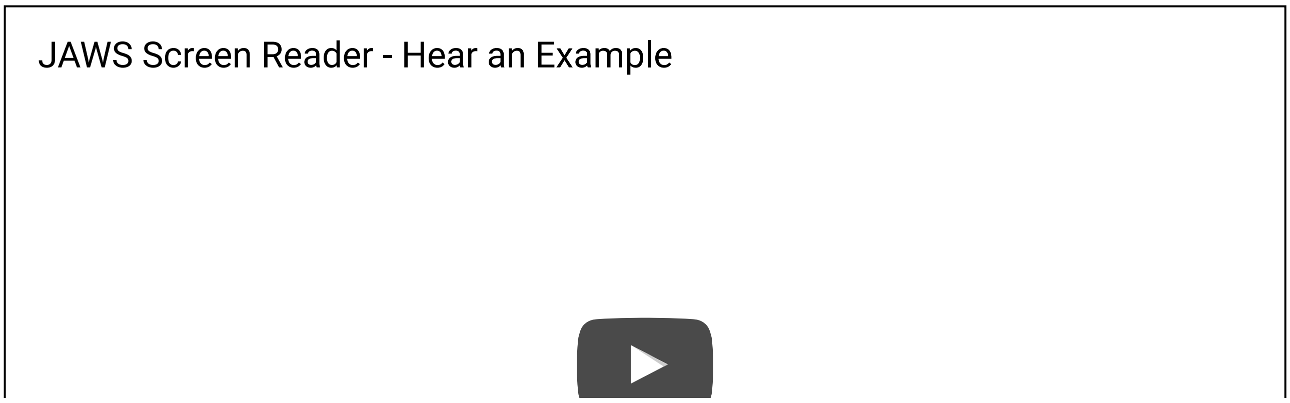
> **Note**: The World Health Organization's Disability and health fact sheet states that "Over a billion people, about 15% of the world's population, have some form of disability", and "Between 110 million and 190 million adults have significant difficulties in functioning."

People with visual impairments

This includes people with blindness, low-level vision, color blindness, etc. Many of these people will use screen magnifiers (either physical magnifiers, or software zoom capabilities — most browsers and operating systems these days have zoom capabilities), and some will use screen readers, which is software that reads digital text aloud:

- Some are paid commercial products, like JAWS (Windows) and Window Eyes (Windows).
- Some are free products, like NVDA (Windows), ChromeVox (Chrome, Windows and Mac OS X), and Orca (Linux).
- Some are built into the operating system, like VoiceOver (Mac OS X and iOS), Narrator (Microsoft Windows), ChromeVox (on Chrome OS), and TalkBack (Android).

It is a good idea to familiarise yourself with screen readers; you should also set up a screen reader and have a play around with it, to get an idea of how it works. See our cross browser testing screen readers guide for more details on using them. The below video also provides a brief example of what the experience is like.



In terms of statistics, the World Health Organization estimates that "285 million people are estimated to be visually impaired worldwide: 39 million are blind and 246 have low vision." (see Visual impairment and blindness). That's a large and significant population of users to just miss out on because your site isn't coded properly — almost the same size as the population of the United States of America.

People with hearing impairments

Otherwise known as people with auditory impairments, or deaf people, this group of people have either low hearing levels or no hearing at all. Hearing-impaired people do use ATs (see Assistive Devices for People with Hearing, Voice, Speech, or Language Disorders), but there are not really special ATs specific for computer/web use.

There are, however, specific techniques to bear in mind for providing text alternatives to audio content that they can read, from simple text transcripts, to text tracks (i.e. captions) that can be displayed along with video. An article later on will discuss these.

Hearing-impaired people also represent a significant userbase — "360 million people worldwide have disabling hearing loss", says the World Health Organization's Deafness and hearing loss fact sheet.

People with mobility impairments

These people have disabilities concerning movement, which might involve purely physical issues (such as loss of limb or paralysis), or neurological/genetic disorders that lead to weakness or loss of control in limbs. Some people might have difficulty making the exact hand movements required to use a mouse, while others might be more severely affected, perhaps being significantly paralysed to the point where they need to use a head pointer to interact with computers.

This kind of disability can also be a result of old age, rather than any specific trauma or condition, and it could also result from hardware limitations — some users might not have a mouse.

The way this usually affects web development work is the requirement that controls be accessible by the keyboard — we'll discuss keyboard accessibility in later articles in the module, but it is a good idea to try out some websites using just the keyboard to see how you get on. Can you use the tab key to move between the different controls of a web form, for example? You can find more details about keyboard controls in our Cross browser testing Using native keyboard accessibility section.

In terms of statistics, a significant number of people have mobility impairments. The U.S. Centers for Disease Control and Prevention Disability and Functioning (Noninstitutionalized Adults 18 Years and Over) reports the USA "Percent of adults with any physical functioning difficulty: 15.1%".

People with cognitive impairments

Probably the widest range of disabilities can be seen in this last category — cognitive impairment can broadly refer to disabilities from mental illnesses to learning difficulties, difficulties in comprehension and concentration like ADHD (attention deficit hyperactivity disorder), to people on the autistic spectrum, to people with schizophrenia, and many other types of disorder besides. Such disabilities can affect many parts of everyday life, due to problems with memory, problem solving, comprehension, attention, etc.

The most common ways that such disabilities might affect website usage is difficulty in understanding how to complete a task, remembering how to do something that was previously accomplished, or increased frustration at confusing workflows or inconsistent layouts/navigation/other page features.

Unlike other web accessibility issues, it is impossible to prescribe quick fixes to many web accessibility issues arising from cognitive disabilities; the best chance you've got is to design your websites to be as logical, consistent, and usable as possible, so for example making sure that:

- pages are consistent — navigation, header, footer, and main content are always in the same places.
- tools are well-designed and easy to use.
- multi-stage processes are broken down into logical steps, with regular reminders of how far through the process you are, and how long you've got left to complete the process, if appropriate.
- workflows are logical, simple, and require as few interactions as possible to complete. For example, registering and signing in to a website is often unneccessarily complex.
- pages are not overly long or dense in terms of the amount of information presented at once.
- the language used in your pages is as plain and easy to follow as possible, and not full of unneccessary jargon and slang.
- important points and content are highlighted in some way.
- user errors are clearly highlighted, with help messages to suggest solutions.

These are not "accessibility techniques" as such — they are good design practices. They will benefit everyone using your sites and should be a standard part of your work.

In terms of statistics, again the numbers are significant. Cornell University's 2014 Disability Status Report (PDF, 511KB) indicates that in 2014, 4.5% of people in the USA aged 21–64 had some form of cognitive disability.

Note: WebAIM's Cognitive page provides a useful expansion of these ideas, and is certainly worth reading.

Implementing accessibility into your project

A common accessibility myth is that accessibility is an expensive "added extra" to implement on a project. This myth actually *can* be true if either:

- You are trying to "retrofit" accessibility onto an existing website that has significant accessiblity issues.
- You have only started to consider accessibility and uncovered related issues in the late stages of a project.

If however you consider accessibility from the start of a project, the cost of making most content accessible should be fairly minimal.

When planning your project, factor accessibility testing into your testing regime, just like testing for any other important target audience segment (e.g. target desktop or mobile browsers). Test early and often, ideally running automated tests to pick up on programmatically detectable missing features (such as missing image alternative text or bad link text — see Element relationships and context), and doing some testing with disabled user groups to see how well more complex site features work for them. For example:

- Is my date picker widget usable by people using screen readers?
- If content updates dynamically, do visually impaired people know about it?
- Are my UI buttons accessible using the keyboard and on touch interfaces?

You can and should keep a note of potential problem areas in your content that will need work to make it accessible, make sure it is tested thoroughly, and think about solutions/alternatives. Text content (as you'll see in the next article) is easy, but what about your multimedia content, and your whizzy 3D graphics? You should look at your project budget and realistically think about what solutions you have available to make such content accessible? You could pay to have all your multimedia content transcribed, which can be expensive, but can be done.

Also, be realistic. "100% accessibility" is an unobtainable ideal — you will always come across some kind of edge case that results in a certain user finding certain content difficult to use — but you should do as much as you can. If you are planning to include a whizzy 3D pie chart graphic made using WebGL, you might want to include a data table as an accessible alternative representation of the data. Or, you might want to just include the table and get rid of the 3D pie chart — the table is accessible by everyone, quicker to code, less CPU-intensive, and easier to maintain.

On the other hand, if you are working on a gallery website showing interesting 3D art, it would be unreasonable to expect every piece of art to be perfectly accessible to visually impaired people, given that it is an entirely visual medium.

To show that you care and have thought about accessibility, publish an accessibility statement on your site that details what your policy is toward accessibility, and what steps you have taken toward making the site accessible. If someone does complain that your site has an accessibility problem, start a dialog with them, be empathic, and take reasonable steps to try to fix the problem.

Note: Our Handling common accessibility problems article covers accessibility specifics that should be tested in more detail.

To summarize:

- Consider accessibility from the start of a project, and test early and often. Just like any other bug, an accessibility problem becomes more expensive to fix the later it is discovered.
- Bear in mind that a lot of accessibility best practices benefit everyone, not just users with disabilities. For example, lean semantic markup is not only good for screen readers, it is also fast to load and performant, so better for everyone, especially those on mobile devices, and/or slow conections.
- Publish an accessibility statement on your site and engage with people having problems.

Accessibility guidelines and the law

There are numerous checklists and sets of guidelines available for basing accessibility tests on, which might seem overwhelming at first glance. Our advice is to familiarize yourself with the basic areas in which you need to take care, as well as understanding the high level structures of the guidelines that are most relevant to you.

- For a start, the W3C has published a large and very detailed document that includes very precise, technology-agnostic criteria for accessibility conformance. These are called the Web Content Accessibility Guidelines (WCAG), and they are not a short read by any means. The criteria are split up into four main categories, which specify how implementations can be made perceivable, operable, understandable, and robust. The best place to get a light introduction and start learning is WCAG at a Glance. There is no need to learn WCAG off by heart — be aware of the major areas of concern, and use a variety of techniques and tools to highlight any areas that don't conform to the WCAG criteria (see below for more).
- Your country may also have specific legislation governing the need for websites serving their population to be accessible — for example Section 508 of the Rehabilitation Act in the US, Federal Ordinance on Barrier-Free Information Technology in Germany, the Equality Act in the UK, Accessibilità in Italy, the Disability Discrimination Act in Australia, etc.

So while the WCAG is a set of guidelines, your country will probably have laws governing web accessibility, or at least the accessibility of services available to the public (which could include websites, television, physical spaces, etc.) It is a good idea to find out what your laws are. If you make no effort to check that your content is accessible, you could possibly get in trouble with the law if people with diabilities complain about it.

This sounds serious, but really you just need to consider accessibility as a main priority of your web development practices, as outlined above. If in doubt, get advice from a qualified lawyer. We're not going to offer any more advice than this, because we're not lawyers.

Accessibility APIs

Web browsers make use of special **accessibility APIs** (provided by the underlying operating system) that expose information useful for assistive technologies (ATs) — ATs mostly tend to make use of semantic information, so this information doesn't include things like styling information, or JavaScript. This information is structured in a tree of information called the **accessibility tree**.

Different operating systems have different accessibility APIs available :

- Windows: MSAA/IAccessible, UIAExpress, IAccessible2
- Mac OS X: NSAccessibility
- Linux: AT-SPI
- Android: Accessibility framework
- iOS: UIAccessibility

Where the native semantic information provided by the HTML elements in your web apps falls down, you can supplement it with features from the WAI-ARIA specification, which add semantic information to the accessibility tree to improve accessibility. You can learn a lot more about WAI-ARIA in our WAI-ARIA basics article.

Summary

This article should have given you a useful high level overview of accessibility, shown you why it's important, and looked at how you can fit it into your workflow. You should now also have a thirst to learn about the implementation details that can make sites accessible, and we'll start on that in the next section, looking at why HTML is a good basis for accessibility.

Overview: Accessibility Next