Progress® Kendo UI®

Accessibility for Web Developers

WHITEPAPER

Table of Contents

Introduction

Accessibility is a topic that has grown rapidly from an afterthought at best to a major concern for UI developers. This has been driven both by the creation and enforcement of laws requiring accessibility features and by a growing awareness of the accessibility needs of the web user community.

At Progress, accessibility has always been an important consideration and we have implemented accessibility features in our Kendo UI® and Telerik® components for some time now. This is an ongoing effort for us as we strive to bring our libraries up to each new release of the WCAG.

Our customers have often had questions not just about how our UI components support accessibility, but about accessibility in websites or apps in general. Some people still have questions about exactly to what extent they need to comply with existing standards, or if the laws even apply to them. The short answer is: completely, and yes they do.

In this paper, we first take a look at accessibility, what it means to web developers and what laws apply. We'll cover different aspects of accessibility and how to implement features that support it. Finally, we'll give a few examples of how to actually implement accessible components along with code examples.

Accessibility does not have to create extra work, and it does not have to restrict what you want to do with your app. As long as you start out with accessibility in mind when you develop your app, it will simply be a built-in feature of your work. You'll stay in compliance with the law and make a lot of users with disabilities much happier.

Let's get started!

1. What is Accessibility, & Why Should You Care?

Every year, more and more of our life is conducted on the web, from shopping and sharing to banking and business. What was done in person, on paper or over the telephone a few years ago all takes place on the web today. We all now have access to a dizzying amount of information and activities online. But as the web has grown in usage, the old ways of doing things have fallen behind.

Paper letters, in-person shopping… these almost belong in a museum now, and, in many cases, there are things that now can only be done on the web and no longer in person. Bank branches and even ATMs have been replaced with online banking. Blockbuster and Borders Books are a distant memory, and Sears filed for bankruptcy after shuttering many stores. Now we have Amazon.

This is great for most of us, and a timesaver for many, but what if you are not able to use the web? What if the store you used to shop at went out of business due to online competition, but you are unable to use the online store? This is where accessibility comes in. And just like ramps on the sidewalk and braille on the elevator buttons have made physical movement easier for those with disabilities, so too have accessibility efforts made the web usable for those who are not able to surf the web in the same way the rest of us do.

It might even surprise some of you that people with disabilities are able to surf the web. But what technology makes difficult, it can also make easy, and there are a number of tools and processes that actually make the web available to pretty much everyone. At least, as long as we follow a few simple guidelines.

Accessibility is often abbreviated “A11Y” because it starts with an “a”, then has 11 letters, then ends with a “y”. And that itself is really a matter of accessibility because if you struggle to enter characters, then a 4-letter word is much better than a 13-letter word. Accessibility is not just one thing either—it’s a collection of things. It’s a little bit user experience, it’s a little bit design, it’s a little bit coding and it’s a little bit testing. But put it all together and you have a pleasant web experience for everyone.

Having said that, let’s stop a minute and think about why we really care about all this. So what if a handful of people have trouble accessing the web? Do we really need to worry about one more thing when we are trying to get our app done? Well, yes, we do. For three main reasons.

There Are 3 Good Reasons to Care About Accessibility

First of all, let's start with being considerate as a reason. It's hard navigating life with a disability, and if you can make people's lives a bit easier with a few changes on your part, then you should do it. You just should, and we shouldn't have to tell you that. So, let's assume you already understand that.

Secondly, and selfishly, it's usually in your best interest. There are more than 56 million people in the U.S. alone with some amount of disability. That's a pretty big market. Do you really want to ignore that many potential customers when all you have to do is use a little more care when you build your sites and apps?

Thirdly, even if you don't care about the first two reasons (which is hopefully not the case), then you need to do it because it's the law. Website accessibility—whether internal or external—falls under the Americans with Disabilities Act (or ADA). And sooner or later, it will catch up to you. In 2017, over 800 federal lawsuits were filed in the U.S. against websites for accessibility violations, up 14 times over 2015. Companies keep trying to argue that the ADA doesn't apply to their website, and almost without exception, the judges keep ruling that it does. So do it now, or wind up in court later.

And there is actually a fourth reason too (yes, we said three—consider this a bonus). If you look at the things that you need to do to make a website or app accessible, most of it is just a bunch of good design practices for everyone. Sure, a deaf person might not be able to hear the soundtrack to your explainer video, but someone in a crowded open office might also just not be able to play it loudly enough to hear. So put in captions—for the deaf user, sure, but also for the user who can't play loud soundtracks. That's just one example of a case where accessibility doesn't just mean being accessible to people with disabilities, it means being accessible to everyone.

Another factor to take into account (but totally not a fifth reason) is that, while people often think that making an app accessible is a lot of extra work, it really isn't. Most of it is simply following good design practices from the start. If you build your app right to begin with, you won't have much extra work at all.

Hitting 100% Accessible

One short but important point to make is that the reality is that you will likely never be 100% accessible to everyone. Accessibility isn't binary, and it isn't measured in "yes" or "no." There are limitations to any approach to providing alternate forms of access. Combinations of disabilities can be extremely challenging to accommodate. And finally, some techniques to help one type of disability can actually make it harder for other types of disabilities.

Accessibility is all about providing the most good for the most people without making your app hard to use for people without disabilities. Fortunately, you don't have to figure this out and there are standards and guidelines to follow to reach this state.

Beauty vs. Accessibility

One final point to list is that making your app accessible absolutely does not mean you have to make it ugly or clunky or violate your company's brand standards. Yes, you may end up making some compromises and creating something different than what you originally wanted. However, a large number of the accessibility guidelines will actually make your app more usable for everyone and provide a better all-around user experience—it just takes a little more work on the frontend to make sure you're compliant.

Next Steps

Now that you understand how disabilities impact a user's access to content and services on the web, the next steps are first taking a closer look at what types of disabilities you need to consider and then examining the laws and guidelines that apply to making the web accessible. We'll then dive deeper into exactly what these mean for your app development and show you how to code popular design elements to guarantee accessibility with actual examples.

2. An Overview of Disabilities

Before we dive into techniques to support accessibility, we should further examine the types of disabilities that we need to accommodate and how they impact a user's ability to access the web. What are the kinds of things we need to worry about? To start off with, there are five primary types of disabilities to consider:

- **Physical** – poor or no motor function
- **Hearing** – deaf or hard of hearing
- **Visual** – low-vision, blind, color blind
- **Cognitive** – dyslexia, autism, ADHD, etc.
- **Speech** – unable to speak or have a speech impediment

In addition, we need to consider combinations of disabilities where possible, which can present even greater challenges. For example, if your solution for users with impaired vision is a screen reader, they still won't be able to use your app or site if they are also hard of hearing.

We also need to consider the platforms that people will be using to access the web. Many people with disabilities use their cellphones to access the web, and this provides a very different experience. We can no longer rely on the keyboard to serve as an input device for those with sight impairments.

For each platform, we need to think about how a user's device affects the functionality of the app. We need to consider non-standard means of interaction: Does my site work without a mouse? Without visual cues? Without sound? It can be very interesting to try to disconnect the individual I/O devices from your computer and see how well your app works. Turn off the monitor, put away the mouse, turn off the speakers… what happens when you do?

To really test out an app, however, we need to test it in the situation that a user with a disability will typically be in. Users with vision impairments will usually be using a screen reader, so while they might not be able to see the screen, it will be read to them. So we also need to consider the use of common assistive tech (AT) that people with disabilities might be using: Does your site work with various AT like readers, magnification, special keyboards, voice-to-text software, etc.? Understanding the environments where they will be used helps create apps that will work well with these devices and also sets the stage for effective testing.

Mobility or Physical Impairments

Mobility or physical impairment refers to weakness, limited ability and inability to independently use one's body or one or more extremities. It ranges from lower body impairments to upper body impairments, and commonly includes:

- Weak / limited muscular control (for instance, tremors, lack of coordination, paralysis, muscular dystrophy, tremors, spasms, repetitive stress injury)
- Limited sensation
- Joint disorders (such as arthritis)
- Pain-impeding movement (like fibromyalgia)
- Missing limbs (such as from amputation)

We usually think of mobility impairments as permanent conditions, but they can also be temporary—when a bone is broken, for example. Keep in mind that many seniors are also prone to mobility impairments.

NOT implementing accessibility impacts the experience of mobility-impaired users in the following ways:

- Inability to access full features and navigate interfaces when full keyword support is not provided
- Not completing tasks because of time limits (like filling out online forms)
- Getting stuck and not accessing the full features of an interface when navigation mechanisms are too complicated or not easily accessible

- Making it painful to use a mouse for extended periods of time when keyboard commands are not enough to navigate

Hearing Impairments

Hearing impairments include:

- Mild to moderate hearing loss (hard of hearing)
- Substantial to uncorrectable hearing loss (deafness)

Something to keep in mind is that there isn't one single sign language, but hundreds of sign languages used throughout the world. Another aspect to consider is that not everyone who suffers from an auditory impairment knows a sign language either.

Today's interfaces increasingly rely on multimedia, so when captions (or transcripts) are not provided, content becomes simply inaccessible for people with hearing impairments.

Also, you should keep in mind that audio content with too much background noise is extremely challenging to understand for people with mild to moderate hearing impairment.

Visual Impairments

Visual impairment covers a wide range of conditions:

- Low vision (short or long sightedness, blurred vision)
- Total blindness
- Color deficiencies

For low-vision users, there are a few no-no's that make their lives really difficult when using a computer:

- Text that cannot be resized.
- Fonts that are too small or too hard to read. Some web fonts have been specifically designed for readability (like Inter UI), so make sure you choose the right one.
- Insufficient color contrast, like text color vs. background color for your buttons. This can mean different things to different users—some users need high contrast, while others need low contrast. This is why it's important to provide adequate markup, so users have multiple options for accessing content.
- Busy page backgrounds, be it patterns or even just too many ads on the page.
- Solely relying on color to convey meaning.

Many people just assume that blind people aren't able to use the internet, when in fact there are ways that enable them to use a computer quite effectively. Namely, a screen reader or braille display. The latter is more appropriate for deaf-blind users, but also more practical for our blind programming comrades.

A keyboard with a braille display: Refreshable braille allows users who are blind or have low vision to use a keyboard like they would paper braille.

However, the following things make it complicated to rely on the tools they have:

- Not providing appropriate alternate text to images.
- Poorly marked-up pages.
- Relying on the use of Java or Flash functionalities.
- Mouse-only website navigation. A mouse requires hand-eye coordination, so if the person is using a screen reader or has limited motor function in their hands, the interface becomes useless when keyboard commands are not enough to navigate.

Some of us think that color blindness means only seeing in black and white, but that is just another misconception. Note that there are different forms of color blindness that encompass the inability to:

- Distinguish between red and green
- Distinguish between yellow and blue
- Perceive any color

Color blind people do not require extraordinary accessibility functions, but there are two things (also shared with low-vision users) that make it very hard for them to navigate the web:

- Insufficient color contrast, like for text vs. background color on buttons
- Solely relying on color to convey meaning

And, by the way, here is a fun fact: Do you know why Facebook is blue?

According to The New Yorker, Zuckerberg is red-green color blind, which means the color he can see best is blue. That also happens to be the color that dominates the Facebook website and mobile app. “Blue is the richest color for me,” he told the magazine.

Source: Why does Facebook have a blue color scheme?

Cognitive, Learning and Neurological Disabilities

Cognitive, learning and neurological disabilities impact how a person:

- Hears and/or understands information
- Moves
- Sees
- Speaks

Some of these disabilities include:

- Attention deficit hyperactivity disorder(ADHD)
- Mental health impairments (anxiety, delirium, depression, paranoia, schizophrenia, etc.)
- Learning disabilities that impact reading (dyslexia), writing (dysgraphia), processing numbers (dyscalculia) or space-time orientation
- Short- or long-term memory loss (caused by dementia, for instance)
- Autism spectrum disorders (autism, Asperger’s, pervasive development)
- Down syndrome that causes learning impairment
- Seizure disorders (such as epilepsy and migraines)
- Multiple sclerosis, which causes nerve damage to the brain and spinal cord extending to other motor and sensory functions

There is this strange assumption that people with cognitive, learning or neurological disabilities are somehow less intelligent or do not need tech features like those without disabilities. For you, hopefully it may seem ludicrous to think like this, but in many communities, this thinking is very common.

In an era in which technology is here to help people advance in life and bring value to one another, technology is a right and inaccessibility is discriminatory. Inaccessibility can limit a person’s potential.

That being said, from mild to life-changing cognitive or neurological disorders, everyone's needs should be accounted for. Things like the issues below add additional barriers to those who have these disorders:

- Uneven gaps between words, very small font and pure white page backgrounds
- Unstructured or verbose content
- Unpredictable functionalities and inconsistent labeling
- Poor navigation options or complex navigation mechanisms
- Distracting content, overly stimulating visual flickering and audio signals at certain frequencies or patterns

Speech Disabilities

We usually don't think of speech disabilities as being a problem for web interfaces, but that is changing as speech recognition is becoming better and better. Digital assistants like Alexa and Siri rely on vocal commands, and any sort of speech impediment can dramatically impact their ability to function at all. What's more, internet-based services and products are increasingly being integrated with these digital assistants, so while your internet-accessible light switch might not require vocal commands to work, the minute you integrate that with Alexa or one of the other voice-based interfaces, you have excluded a portion of the population from access. This is fine, as long as it's simply an extra way to control those systems—but don't forget that not everyone can use this integration.

The other consideration is where people might have multiple disabilities. If your user is vision impaired, you might have a good alternate input using speech-to-text. But what do you do if the user also has a speech impairment? Supporting multiple disabilities can be challenging, but it is something to consider and take into account as much as possible.

Situational Disabilities

We have been considering people with typical disabilities that we are largely familiar with, but there is another class of users that we should include in this grouping: people with transient or situational disabilities.

First of all, we need to include people who might not have a very good handle on technology. This group is largely older users who grew up without computers or the internet. Many of them simply are not comfortable or particularly familiar with the use of technology. Add to that age-related issues like reduced eyesight, hearing or mobility on top of simply not understanding what a user interface is is asking, as these factors can have the same effect as an actual cognitive impairment.

Next, we should consider the environment that the user is working in. What if they are surfing the web in a noisy Starbucks, left their earbuds at home and just can't hear the nifty explainer video you posted? What if they are in a sunny park with their laptop and the bright sun is washing out their screen and making it hard to see? Or what if they are in a location with very slow Wi-Fi (the horrors!) and each page is loading very slowly?

In addition to location, consider equipment. What if they are on a mobile phone, an old computer or a computer with an old version of a browser?

And finally, what about temporary health issues? You would not expect someone to be buying rock climbing gear if they did not have the use of their hands, so maybe you don't account much for physical disabilities on your rock-climbing store site. But what if they simply broke their hands on their recent rock-climbing excursion and, because they are slow learners, want to buy some more equipment so they are ready when the cast comes off?

These are situations where it is dangerous to assume that you know your users and are certain that they will not be operating with specific disabilities.

3. Accessibility and the Law

This is the part where we issue the disclaimer that we are not a law firm, and this is not legal advice. This is a collection of information that we have found while working with accessibility. Please consult with an actual lawyer if you have any specific legal questions.

The web continues to evolve from merely providing useful but essentially static information to a place where dynamic applications are widely present. This shift has spread to all aspects of our society, and we now have online commerce (Amazon... need we say more?), online banking, online education (from product training to actual degree programs), online government (services, communication, applications, even voting) and more, including the delivery of basic communications and services.

Global Awareness

Along with this evolution has come a general awareness of the need to structure these services in such a way that they can be used by those with various disabilities. Laws and guidelines have been put in place by worldwide organizations like the United Nations as well as individual governments and even industries.

In 1982, the General Assembly of the United Nations adopted the World Programme for Action (WPA) concerning Disabled Persons. This was a global strategy to enhance disability prevention, rehabilitation and the equalization of opportunities. The key here is the "equalization of opportunities."

The Standard Rules on the Equalization of Opportunities for Persons with Disabilities, adopted by the UN in 1993, specifically added that the member states' action programs should include:

"Support for the use of new technologies and the development and production of assistive devices, tools and equipment and measures to facilitate access to such devices and equipment for persons with disabilities to enable them to gain and maintain employment."

These actions both reflected and helped drive further awareness of the need to provide access to technology, although at that time the full range of opportunities with the internet was still in the future.

Country and State Accessibility

A number of standards have been created that describe policies and procedures to use when making the web more accessible. Most of these were sponsored by various government entities, but the most widely used came from the Web Accessibility Initiative (WAI), which is part of the World Wide Web Consortium (W3C). The W3C is an international organization dedicated to developing open standards for the web.

The W3C lists 40 countries that have adopted governmental policies related to web accessibility, and this is not even a complete list. Many use standards that they have developed, but many also have standardized based on the Web Content Accessibility Guidelines (WCAG), which is the standard developed by the WAI. The WCAG has been approved as an international standard by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). Version 2.1 of WCAG was released in June of 2018, and although it is by itself a voluntary standard, it has been referenced by a number of laws on accessibility.

The United Kingdom, for example, passed the Equality Act of 2010 which is a non-discrimination law that applies to both the public and private sections and is similar to the American ADA. The law is a broad non-discrimination law that replaces a number of earlier individual non-discrimination laws. It covers discrimination based on age, gender, race, religion, sexual orientation and, of course, disabilities. This law does not directly specify the WCAG (or any other standard), but the U.K. Government Service Manual's section on accessibility and assisted digital does specifically require WCAG 2.1 compliance for all government websites.

Note that the WCAG 2.0, released in 2008, introduced the important concept of three levels of conformance. These are:

Level A: Some segment of the population will find some or all of the web content to be completely inaccessible

Level AA: Some segment of the population will find some or all of the content to be difficult to use

Level AAA: The content is as accessible as reasonably possible with the current state of web content and/or technology

For organizations to claim compliance at any specific level, they must meet all of the requirements for that level and all levels below it. At this time, meeting Level AAA is not expected nor required.

Web Accessibility in the United States

In 1990, the U.S. enacted the Americans with Disabilities Act (ADA) to prohibit discrimination based on disabilities. This was an extension to the Civil Rights Act of 1964, which outlawed discrimination based on race, religion, sex, national origin and other characteristics. The ADA made it also illegal to discriminate based on disability. What's more, it required employers to provide reasonable accommodations to employees with disabilities, and it places accessibility requirements on public accommodations. The bill was later amended in 2008 and went into effect in 2009.

In addition to the ADA, there are a number of other laws in the U.S. that cover accessibility in various circumstances, including several sections of the U.S. Rehabilitation Act of 1973 as amended, Section 225 of the Communications Act, the 21st Century Communications and Video Accessibility Act of 2010, and more. In addition to that, a number of states have passed laws concerning accessibility as well. If that all sounds confusing, that's because it is. However, the WCAG is generally considered to meet or exceed all of these requirements. And this is a good time to remind you that this is not legal advice and you should seek legal counsel if you have questions.

The other standard that you might hear about is Section 508 of the Rehabilitation Act mentioned above. The Rehabilitation Act was amended in 1998 to require federal agencies to make their electronic and information technology (EIT) accessible to people with disabilities and was further updated in 2018 to meet evolving standards. Section 508 is now based on the WCAG 2.0 guidelines.

The Law in Action

We've taken a quick look at some of the laws that have developed concerning accessibility and websites. This can be confusing and unless you really know what you are doing in this area, it's a very good idea to engage the services of a company that does have accessibility expertise to at least help train your dev team and review your work. In general, conforming with the WCAG will be a good start. But do you really need to worry about it, especially if you are not doing business with the government? The answer is a very clear "yes" as a number of recent legal cases have shown.

According to a search of court cases conducted by Minh N. Vu, Kristina M. Launey & Susan Ryan of law firm Seyfarth & Shaw, the number of lawsuits concerning website accessibility nearly tripled from 2017 to 2018 and hit a total of 2,258 cases. Some of these cases have received public attention as well and highlight both the growing need to adhere to accessibility standards, and also the ongoing work that needs to be done in establishing clear rules.

A recent case involving Domino's Pizza saw the Ninth Circuit Court of Appeals issue a ruling that overturned a district court's dismissal of a suit in the favor of accessibility. The Ninth Circuit held that the ADA applies to services of a place of public accommodation, not just services in a place of public accommodation. Domino's customers access the website away from the physical location (the pizza shop). The initial suit claimed that the Domino's website did not work with the customer's reader software, so he could not order a pizza from them. This was argued to be a violation of the ADA. The Domino's website was weak in accessibility, and that limited access to the company's pizza services.

Note that this is not the end of the Domino's case. The Ninth Circuit's ruling simply means the suit can continue now. But it is still a victory for accessibility—and, in fact, sets a precedent that companies can legally be sued for digital accessibility noncompliance.

4. Assistive Technologies

Assistive technology is defined as a piece of equipment or a type of product designed to improve or maintain the functional capabilities of people with disabilities. Assistive technologies can make a world of difference for those with disabilities, helping them lead comfortable, productive lives wherein they're able to take part in various activities that would otherwise be difficult. There are various assistive technologies available, including:

- Tools, such as adapted pencil grips or automatic page turners, designed to improve the outcome of educational tasks
- Cognitive aids, such as electrical assistive devices, created to help memory skills

- Mobility aids, such as wheelchairs or scooters, designed to help improve the individual's ability to move

In terms of web accessibility, assistive technologies can be used to help ensure the individual is able to leverage the internet and any web-related content. Some examples include:

A mouth stick: A simple device that users can manipulate with their mouths to type or control a mouse

- A head wand: A simple device strapped to the user's head that enables them to type, navigate and otherwise maintain control over web services
- Oversized trackball mouse: A device that lets users move their cursor easily, despite any fine motor movement limitations
- Adaptive keyboard: A popular technology featuring raised areas in between keys to help slide the finger into the correct key
- Voice recognition software: Software that enables users to control web services with simple voice commands

One example of a popular assistive technology tool is JAWS from Freedom Scientific, which is a Windows screen reader. A screen reader is a software program that enables a blind or visually impaired user to read the text that is displayed on the computer screen with a speech synthesizer or braille display. JAWS (Job Access With Speech) was developed for computer users whose vision loss prevents them from seeing screen content or navigating with a mouse. JAWS provides speech and braille output for the most popular PC applications. There are a variety of free screen readers as well as other commercial screen readers, but JAWS is one of the most popular.

There are also a number of hardware-based tools that function as assistive technology. A good example is the Xbox Adaptive Controller from Microsoft. This device was designed to meet the needs of gamers with limited mobility. In addition to providing easy-to-use input devices, it also serves as a hub for a variety of external devices which gives the user great flexibility in how they configure the device. The user can connect devices like switches, buttons, mounts, joysticks and more to create a custom controller environment.

To develop the Xbox Adaptive Controller, Microsoft worked directly with groups like the The AbeGamers Charity, The Cerebral Palsy Foundation, SpecialEffect, Warfighter Engaged and accessibility community members. Engaging with subject matter experts when creating any accessible technology is extremely important to make sure the technology will actually help the target community.

5. Digital Accessibility Requirements

Accessibility requirements (or, as WCAG calls them, "guidelines") can present a real learning curve for the uninitiated. The guidelines are abundant and often vague. It can be hard to know how to take action toward accessibility compliance unless you have significant accessibility expertise. That's partially because WCAG organizes accessibility guidelines into four major principles that define what the content on your site or app must be in order to be usable for those with disabilities:

1. **Perceivable**

   Is the information presented in a way that people can access and perceive it with at least one of their senses?

2. **Operable**

   Can users navigate and find content, and can people use your interface without a mouse? Are they given plenty of time to complete tasks and read content? Is the content safe from causing a seizure or physical reaction?

3. **Understandable**

   Can people understand both your information and how to operate your interface, no matter how they're accessing your content?

4. **Robust**

   Can all kinds of user agents and assistive technologies interpret your interface for those using it? Will you evolve your content as assistive technology evolves to ensure new technologies can interpret it for those trying to access it?

These principles are a good guiding overview as you work to make your site, application or software suite accessible. However, they can also be pretty abstract if you're new to web accessibility and taking your first steps to implement accessibility improvements. What specifically does it mean for digital content and interfaces to be perceivable? Operable? Understandable? Robust? These words probably don't mean much to you unless you already really know your way around the accessibility guidelines.

We'll address accessibility guidelines instead by common types of issues and how to fix them.

While there's no way we can cover every single guideline here, we can address some of the most basic accessibility issues that give organizations the most trouble. In fact, you probably think you're following some of these requirements. An accessibility scanner might even tell you that you are. But there's a big difference between passing an automated scan and creating digital products that people with disabilities can actually use. It's very likely that even the accessibility requirements you think you're following are issues plaguing your interface at this very moment.

What follows are some of the largest barriers to digital accessibility.

Alternative Text

We'll start with alt text because it's the most commonly known accessibility requirement. Most people know they should write alternative text for images (even if they don't do so in practice). However, the sheer abundance of poorly written and inaccessible alt text out there would suggest that most of us are not aware of what it takes to write descriptions of images that will be at all useful for people with disabilities.

The WCAG Guideline

Guideline 1.1 says, "The purpose of this guideline is to ensure that all non-text content is also available in text." Any images that add context or information to a page must have alt text describing that information.

Guiding principle: This guideline falls under WCAG's principle that online content and interactions must be "perceivable."

What This Means for Users with Disabilities

People using screen readers to help them navigate your interface will be able to hear the alt text you wrote to understand the image. Without this alt text, there is no way for blind and low-vision users to get the information found in the image. Screen readers are also sometimes used by people with learning disabilities to help them receive information through multiple senses.

Best Practices

1. Know when to use alt text

Images require alternative (alt) text if they:

- Convey information that isn't available elsewhere in the text on the screen
- Help the user understand the content

- Contain an action (e.g. you click on the image and something happens)
- Contain text that can't be found on the page

If an image doesn't do any of the above, then it is considered decorative, and the alt text should be left blank.

2. Describe the spirit and context of the image

Helpful alt text conveys the same information in words that the image is conveying visually.

Ask yourself:

- What is important about the image?
- What information is the image adding to the page?
- How would you relay that information to someone who couldn't see it?

3. Include any text that is part of the image

Screen readers cannot see text that is embedded inside of an image. Therefore, helpful alt text should repeat whatever text is in the image. Better yet, you could take the text out of the image and present it as live text on the page.

4. Describe any action related to the image not already described in surrounding words

If the image is being used for an action (the user clicks on the image and something happens) and there are no accompanying words on the page to describe the action, helpful alt text will present the action that will occur if the user clicks the image.

5. Avoid using "Photo of" or "Image of" in alt text

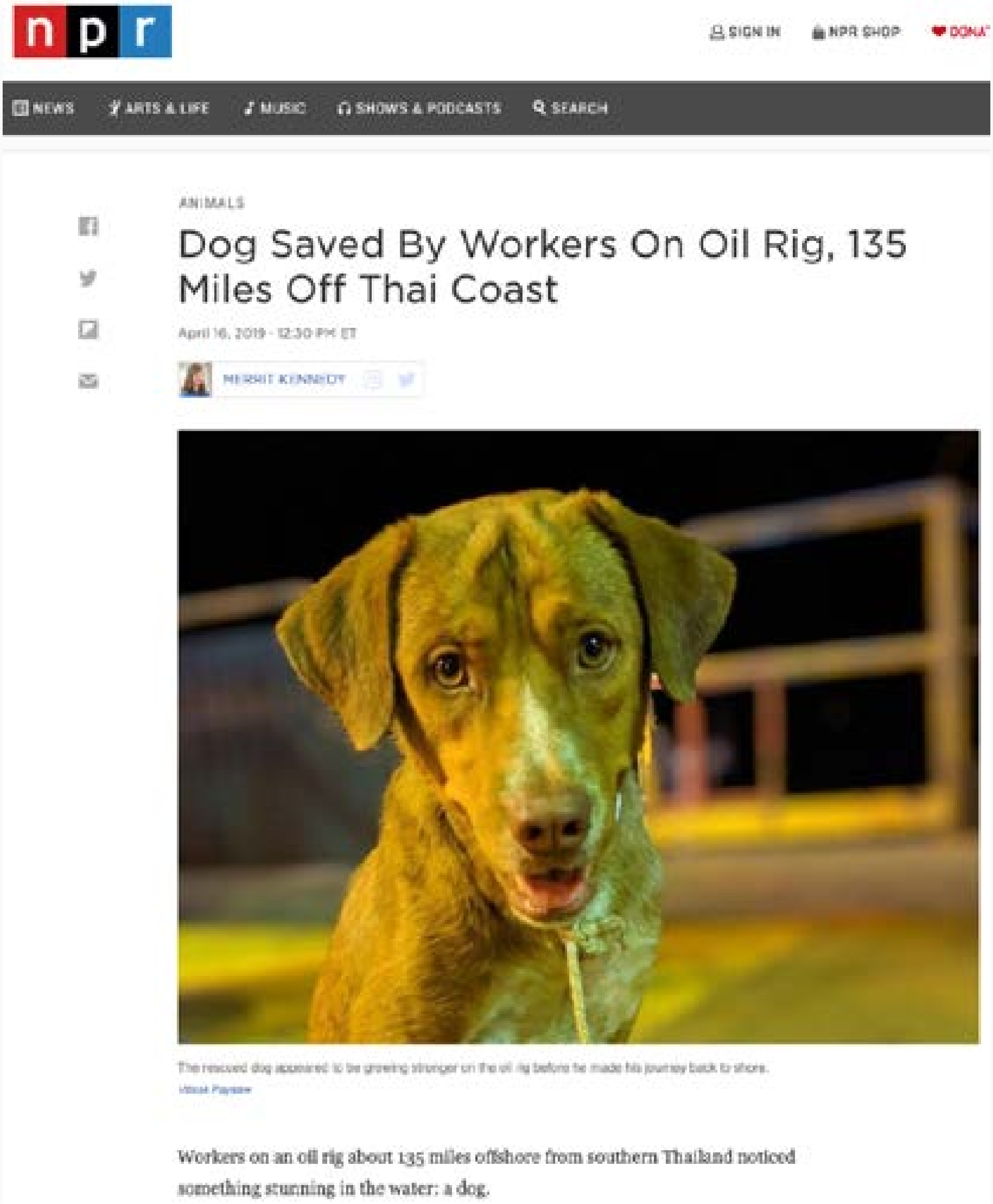
Screen readers already tell users when something is an image, so including that information in the alt text is redundant and obnoxious for users.

6. Be as succinct as you can

Convey the meaning of the visual in as few characters as possible, while still communicating the full intent of the image. Users with screen readers have to listen to every word you write, so anything extra can cause annoyance and frustration.

7. If an image doesn't offer additional information not found in words on the page, leave it blank

If your image truly doesn't present any information that can't be gathered another way, you can leave the alt text blank so the screen reader will skip the image entirely.



ANIMALS

Dog Saved By Workers On Oil Rig, 135 Miles Off Thai Coast

April 16, 2019 · 12:30 PM ET

MERRIT KENNEDY

The rescued dog appeared to be growing stronger on the oil rig before he made his journey back to shore.

Vitsuk Payakaw

Workers on an oil rig about 135 miles offshore from southern Thailand noticed something stunning in the water: a dog.

Alt text example: This photo conveys information about the dog not found elsewhere in the article. Because the photo has a caption describing that extra information, it could be reasonable to leave alt text blank for this image. If the caption were removed or didn't mention something communicated in the photo, however, the image would need alt text.

This story from NPR includes photos of a dog that was rescued at sea by oil rig workers. The photos of the dog present visual information not mentioned in the rest of the article. However, these photos have captions, like the one above, that explicitly describe what is happening in the photo. For that reason, alt text for these photos could reasonably be left blank. If the captions were removed or if they left out important information found in the photo, these images would need alt text.

Helpful alt text: "Happy-looking, clean dog on the deck of an oil rig wearing a homemade braided rope leash."

Unhelpful alt text: "Photo of dog"

Page Structure

Think of the most horribly disorganized, long-winded site or app you've used recently. For users with disabilities, that's every page that doesn't have the correct best practices in place for page structure, but times 1,000. If a page is difficult to use for those who don't require any assistive technology, it will be monumentally difficult to use and time-consuming for those who do. Excellent page structure is everything for those navigating with a keyboard or a screen reader.

To be useful for those with disabilities, the structure of a page and the hierarchy of elements on it must be conveyed both visually and semantically. In other words, the code used on a page should also reflect the visual structure of the page and elements within it.

The WCAG Guidelines

Guideline 1.3 requires that "information, structure and relationships conveyed through presentation can be programmatically determined or are available in text."

Success Criterion 1.3.1 to meet Guideline 1.3 requires providing "programmatic access to sections of a web page," marking headings so "they can be programmatically identified" and generally using semantic elements so content is clearly defined.

Guideline 2.4 requires "ways to help users navigate, find content and determine where they are." Headings, labels and the way you structure the page all contribute to meeting this guideline.

Guiding principle: These guidelines fall under WCAG's principles that online content and interactions must be "adaptable" and "navigable."

What This Means for Users with Disabilities

The only way for those using a screen reader to know what's on the page and navigate to different parts of the page without having to listen to every single word is through a clear, well-organized semantic structure. People using screen readers are able to navigate through a page using a number of methods. They can jump between sections of a page using ARIA landmarks or HTML5 elements like HEADER, MAIN and FOOTER. They can quickly skim content by using headings and subheadings to navigate and tab through links and buttons to navigate quickly through the site or application. That is, if all of these things are available to them on your site or app.

Best Practices for Code

Convey important elements on the page using appropriate HTML5 elements or ARIA landmarks. Never rely on visual cues to inform the user of a page element's purpose. Following coding best practices will almost always result in a well-structured page. Use common sense here. If it quacks like a duck and looks like a duck, call it a duck. Do not call it anything similar or synonymous or clever, like "water chicken."

- If a list of items is used to navigate, make sure it's within the NAV element.
- Main navigation should appear in the HEADER element despite it not being at the top or "head" of the page.
- Lists, regardless of how they are displayed visually, should always use OL, UL and DL elements.
- Table markup should be used for tabular information and marked appropriately when only being used for presentation purposes.
- Clickable elements should always be semantically described as links, buttons, or form controls. Using visual cues to make a traditionally non-interactive element appear clickable won't help those using screen readers.

Best Practices for Headings

1. Assign each heading and subheading a tag, H1 through H6
Avoid using bolded paragraph text for smaller subheadings—screen readers do not recognize bolded text as something that indicates a navigational section. Instead, create a heading the same size and weight as your bolded text named H5 or H6.

2. Nest subheadings in order
Use headings and subheadings in rank order (H2, H3, H4, etc.). Headings can be reused, resetting their order for each section or landmark on the page. Within using different subheads within a single section or landmark, be sure to follow the correct numerical progression.
For example, avoid placing an H4 heading immediately after an H2 heading, skipping H3 in the progression. This can be incredibly confusing for those who cannot understand your information hierarchy from visual cues.

3. Write clear, simple headings
Remember that your headings serve as a navigational tool. It should be easy to understand exactly what information will be under a given heading or subheading. Once again, if it's a duck, call it "Duck."

Color Contrast

The general rule of thumb with color contrast is that it must be easy to differentiate text from its background and links must clearly stand out from non-linked text. But sometimes that's much more difficult than it sounds.

Color contrast is an area of accessibility where many websites royally fail in their compliance efforts. Many organizations' brand palettes are not created with online accessibility in mind, so when you try to use those colors on a screen, the result is often poor color contrast. This is also one of the areas of accessibility that can cause the most heartache because, in some cases, it means tweaking your established brand colors for your site or app to meet accessibility compliance. Not a fun sell to the marketing and communications departments.

The WCAG Guidelines

Guideline 1.4.1 requires that "color is not used as the only means of conveying information, indicating an action, prompting a response or distinguishing a visual element."

Guideline 1.4.3 requires that "the visual presentation of text and images of text has a contrast ratio of at least 4.5:1."

Guiding principle: These guidelines fall under WCAG's principle that online content and interactions must be "distinguishable."

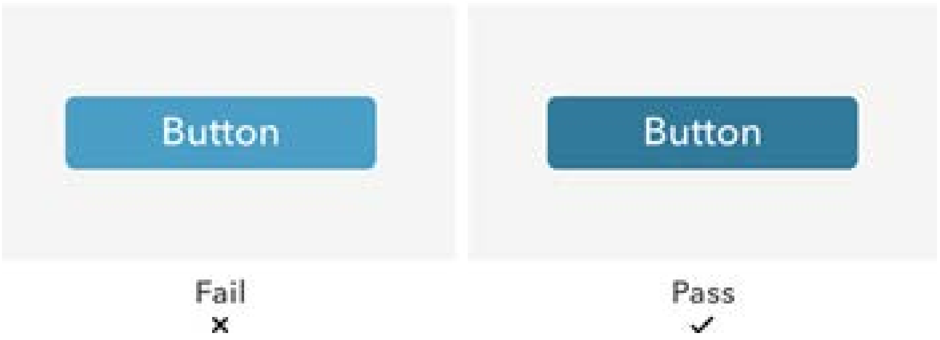
What This Means for Users with Disabilities

People with low vision or who cannot see colors well (those with color blindness, for example) cannot read text or identify links if the foreground color is too close to the background color. And even if your contrast between the text and the background is good, you should still never assume that color will be enough to help a user identify links.

Best Practices

1.Check color contrast for all text
Use a tool like WebAim Color Contrast Checker to see if your text falls into compliance. The

minimum contrast for regular text is 4.5:1 unless you are using large text (at least 18 pt or 14 pt bold). For large text, the minimum contrast is 3:1.

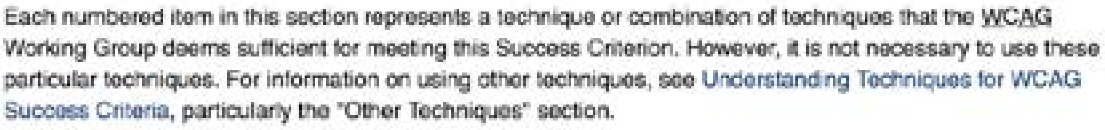


Button contrast comparison: The light blue on the left (the organization's brand color, by the way) did not meet color contrast standards. When darkened slightly, the blue passes.

2.Add a non-color design element to identify text links

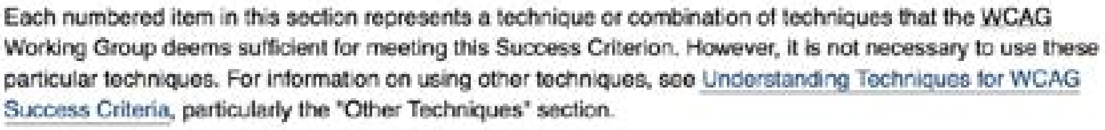
Identify links in a way that does not rely solely on color. Options include:

- Underlines
- Carats
- Icons



Each numbered item in this section represents a technique or combination of techniques that the WCAG Working Group deems sufficient for meeting this Success Criterion. However, it is not necessary to use these particular techniques. For information on using other techniques, see Understanding Techniques for WCAG Success Criteria, particularly the "Other Techniques" section.

Link identification relies on color: This link within the paragraph would not pass contrast compliance because the user must rely on color differentiation alone to identify the link.



Each numbered item in this section represents a technique or combination of techniques that the WCAG Working Group deems sufficient for meeting this Success Criterion. However, it is not necessary to use these particular techniques. For information on using other techniques, see Understanding Techniques for WCAG Success Criteria, particularly the "Other Techniques" section.

Link can be identified without color: This link within the paragraph passes contrast compliance because even if the user cannot differentiate the blue link color from the black text, the underline also indicates a link.

Focus States

Many a digital product is guilty of focus states that are too weak to help people with disabilities understand what element they're focusing on or selecting. This can cause much frustration for users when they click the wrong thing or when the thing they thought they clicked doesn't go to the right place.

When an element has focus, it must be visually obvious to the user.

The WCAG Guidelines

Guideline 2.1 requires that "all functionality of the content is operable through a keyboard interface."

Guideline 2.4.7 says that the keyboard focus indicator must be visible in these cases.

Guiding principle: This guideline falls under WCAG's principle that online content must be "navigable."

What This Means for Users with Disabilities

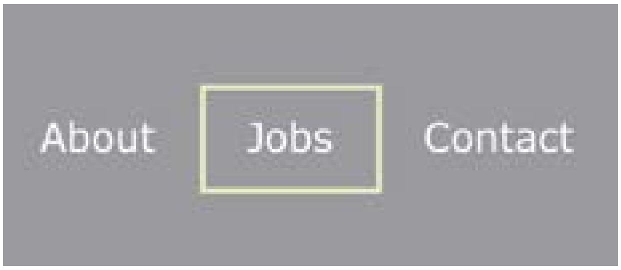
People who are unable to use a mouse rely on either their keyboard or an assistive device. Usually this means they have to navigate by moving from one element on a page to the next in some kind of sequence. Focus states enable users with disabilities to visually determine which element they are on. From there, they can choose to interact with that element or move onto the next one.

Best Practices

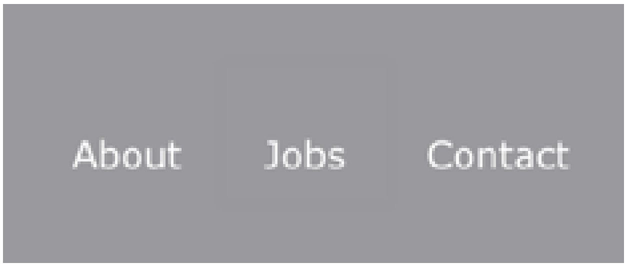
1. Don't remove focus states
By default, most browsers will display an item's focus state. Sometimes, developers purposely or accidentally disable this state through scripting or styles. This makes it impossible for someone using a keyboard to see what element they are about to select.

2. Don't rely on default focus states
The browser's default focus state might not always be very noticeable. Implement custom focus states that fit the overall design while still being easily noticed.

Focus State Examples



Accessible focus state: This custom focus state clearly indicates what a user is about to click with a visible yellow outline.



Inaccessible focus state: This default Google Chrome focus state is light gray and therefore almost impossible to see on this gray background.

Video and Audio

People with disabilities want access to the same video and audio that everyone else enjoys every day. Imagine being unable to enjoy the TV shows, movies or podcasts that play a vital part in popular culture just because no one had thought to make sure the information was accessible to you. Users with disabilities can enjoy the same media as everyone else, they just need to access it in a different way.

The WCAG Guidelines

Guideline 1.2 says, “Provide alternatives for time-based media.” Here, time-based media means anything video- or audio-related.

Guideline 1.4.1 requires that “if any audio on a web page plays automatically for more than three seconds, either a mechanism is available to pause or stop the audio or a mechanism is available to control audio volume independently from the overall system volume level."

Guiding principle: These guidelines fall under WCAG's principle that online content and interactions must be "perceivable."

What This Means for Users with Disabilities

Users who are deaf or hard of hearing and users with learning disabilities rely on captions and transcripts to access audio information. Alternatively, blind users need audio descriptions to relay a video's visual information. Video and audio that play automatically can cause issues for those using screen readers. Audio from the screen reader becomes harder, if not impossible, to understand when audio from a video overlaps it.

Best Practices

How do you make sure your video and audio content is accessible?

1. Allow videos to be paused

2. Allow audio to be turned off

3. Provide audio descriptions for videos

4. Provide captions or a transcript for videos and audio

Forms

Digital forms are often a huge challenge for users with disabilities. We all know how important these forms can be in everyday life. Most of us use them without even thinking about it to complete critical tasks like applying for jobs, paying bills and managing our ever-growing number of online accounts. Imagine if every time you went to do one of those crucial tasks, you were faced with hours of effort just to figure out how to complete the form. That's very often the reality that people with disabilities live in.

To make accessible forms that don't horribly frustrate your users with disabilities, all fields must be clearly labeled, success and error messages must be obvious and all fields need to work with a keyboard.

The WCAG Guidelines

Guideline 3.3.2 requires that “labels or instructions are provided when content requires user input.”

Guideline 4.1.2 says that, for all components, “the name and role can be programmatically determined.”

Guideline 3.3.1 says, “If an input error is automatically detected, the item that is in error is identified and the error is described to the user in text.”

Guiding principle: Guidelines for forms fall under every one of WCAG’s principles for online content: “perceivable,” “operable,” “understandable” and “robust.”

What This Means for Users with Disabilities

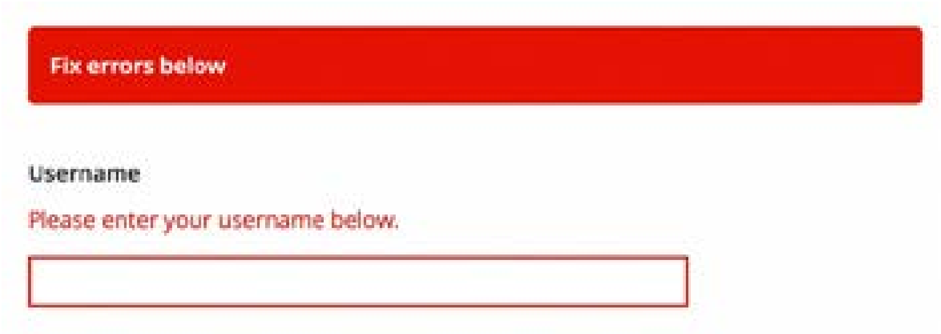
People who use a keyboard to navigate forms rely on being able to get to fields in an organized, sensical manner. They also need to be able to see clearly which field or component they’re about to fill out.

People who use screen readers to navigate digital forms rely on programmatically set form labels to provide instructions and help them understand a component’s purpose. Users who are blind or have low vision also cannot use visual-only cues for errors and success messages. Visual indicators that mark required or incorrectly entered fields don’t mean anything to someone who can’t see them. Likewise, success messages that appear and can be read by a screen reader after a form has been submitted are important so users don’t attempt to submit the form again. Submitting a form multiple times can be really problematic if you’re, say, paying your credit card bill or applying for a loan.

Best Practices

1. Label and group controls
All form controls must be labeled, and related controls must be grouped. Grouping related controls in a FIELDSET with a legend makes forms much easier to understand and fill out. Similarly, putting a label before a control makes it much easier for users to associate the two.

2. Notify users of error/success of form submission.
You must notify users clearly of the result of their form submission, whether it was successfully submitted or if there are errors they need to correct. Ideally, keyboard focus should immediately be given to a global error or success message at the top of the form after the user submits it. This will help users quickly understand what they need to do next.



Fix errors below

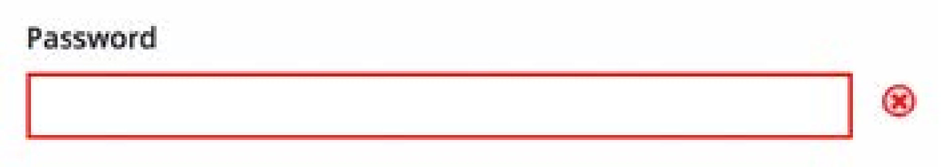
Username

Please enter your username below.

Global error message: Displaying a global error message above forms when there are multiple errors makes it easier for users to understand that there are items that need to be fixed.

3. Utilize control states

Never rely on visual cues alone to inform the user of errors or that a field is required. These won't work for users who are blind or have low vision. Instead, use the "required" attribute on controls that need it and convey errors in text that is easy to find on the page.



Password

Visual-only error cue: Users who are blind or have low vision won't see the red outline or icon next to this field. Text should display under the label explaining the error for a screen reader to read.

PDFs

PDFs are perhaps the biggest beasts of digital accessibility. Depending on how they were created, they can be time-consuming and difficult to remediate. PDFs must either be accessible themselves, or the information found in the PDF must be available in another, accessible way on the site or app. And when we say "information," we mean all content found in the PDF, including charts, graphs, images, infographics, etc. Think of the most complicated, graph-heavy PDF your organization or client's organization has (maybe an annual report or investor brochure) and you can begin to understand how much work it would really take to ensure the document is accessible.

The WCAG Guidelines

PDFs have a huge number of guidelines because they must follow the rules for any other page PLUS adhere to WCAG's "PDF Techniques." Basically, depending on the PDF, you may need to comply with any and all WCAG guidelines to make the document accessible. You can see how remediation of a single PDF can take a long, long time.

Guiding principle: Because PDFs must comply with every WCAG guideline, every guiding principle also applies to making PDFs accessible. PDF content must be "perceivable," "operable," "understandable" and "robust."

What This Means for Users with Disabilities

PDFs are often extremely problematic for users who are blind, have low vision or navigate using a keyboard. If a PDF is entirely inaccessible for those with visual disabilities, for example, a screen reader may tell a user that the PDF is blank, even when they know for certain it is not. At that point, they have to either give up trying to access the information or call in a favor for someone else to explain the PDF to them.

If a PDF hasn't been tagged with the correct navigational elements, users navigating with a keyboard will have difficulty tabbing through to find the information they need. Especially for long PDFs, this can mean they waste their time tabbing through a document to find the one topic or piece of information they were looking for.

Best Practices

1. Avoid PDFs where possible
As often as you can, turn PDF content into on-page content. It's much easier to make a screen in your site or app accessible than to make a PDF accessible. This practice is much better for overall usability, as PDFs cause difficulties on mobile devices and in search as well.

2. Create accessible PDFs to begin with
It's much harder to remediate a PDF after the fact, especially if you're not the one who originally made the document. Ensuring that your PDFs are accessible from the start will save you a lot of time and headache. Just walking through how to make a PDF accessible could fill its own whitepaper, but there is a lot of good help online. Most document creation software companies also have guides and wizards you can use to ensure your document is accessibility compliant. Here are guides for some of the most commonly used software:

- Microsoft Word Guide
- Adobe InDesign Guide
- Adobe Acrobat Guide

3. Ensure content is available elsewhere on the site
If a PDF is necessary but it's going to prove difficult and/or time consuming to remediate, you may decide to create an accessible, on-page version of the content that is either only visible to

screen readers or is otherwise out of the way unless someone with a disability needs to access it. As long as all of the information found in the PDF is available in another accessible way, you're good to go.

Conclusion

This may sound like a lot, and it is. There are plenty of guidelines and issues you'll come across that we don't have space to address here. The good news is, better accessibility almost always means better overall usability. It will take creative problem solving and maybe even some compromise in the odd but inevitable case where the rules of accessibility and the rules of usability don't agree.

Get started by implementing these best practices for some of the most common and difficult issues. You'll thank yourself when you run an accessible scan or conformance test of your site or app. The number of remaining issues the scan flags or human testers find will be significantly lower for your efforts, reducing some of the strain on yourself and your team.

As long as you're focused on creating a usable, accessible experience for all users, you're on the right track. In the long run, it will help you reach more users, increase engagement and perform better against success measures.

6. Conformance and Testing

The benefits of performing accessibility testing are clear, but where do you start? As web accessibility becomes a more prevalent topic, training programs are popping up all over the place to help teach individuals how to create accessible content.

Most training programs focus on the WCAG guidelines, which provide various technical specifications and techniques for web developers to follow when it comes to creating accessible content. So how do you perform accessibility testing?

Perform Manual Testing

There are a lot of accessibility issues that might be missed with a computer program. Manual testing involves having human testers perform a range of tests to ensure the website is actually usable for individuals with disabilities. Similar to any type of editing or quality assurance, you're able to get an entirely new viewpoint with each person who performs manual testing. Testers will look for:

- Compatibility with various assistive technologies
- Navigation that is simple and easy to understand
- Content that is meaningful and clear/concise
- Coordination with color adjustment plugins for various web browsers
- And much more

Any testers you employ should be familiar with the latest standards when it comes to web accessibility. That means they must understand WCAG to ensure they're looking for the right factors that matter to people with disabilities.

Perform Automated Testing

Automated testing is the process of using software tools designed to find accessibility issues. These tools have evolved quite a bit over the past few years, so it's no longer difficult to find a range of tools available to help you adhere to accessibility guidelines. In fact, there are browser extensions, command-line tools and much more. These tools should be used at the beginning of the design process and periodically throughout to catch issues before they go into production.

Why Use Manual and Automated Testing?

Manual and automated testing should be used in combination with one another to ensure a thorough testing process that doesn't miss anything important. Automated tools are great for finding:

- Images without alt text
- Content that doesn't have headings
- HTML code that isn't valid
- Form fields that are missing labels and/or descriptions
- And much more

However, as much as technology has advanced, human intervention can help find issues that no software tool is able to pinpoint, such as:

- Alt text that isn't accurate given the context
- Descriptions that aren't clear or easy to understand
- Headings that don't make sense given the hierarchy
- And much more

From a usability standpoint, something like a misleading alt tag makes a world of difference, even though it's not findable with an automated tool. Testing a screen reader can only be done manually, and it often catches issues that would otherwise be missed.

Automated Accessibility Testing Tools

As mentioned, there's a wide range of software tools available for automated testing. The process of testing for accessibility should be ongoing throughout all stages of development. If you have the means, a continuous integration solution can be very beneficial. Some common tools for automated testing include:

Dyno Mapper

Dyno Mapper allows you to test entire websites for WCAG 1.0, 2.0, 2.1/Section 508 compliance, as well as various other guidelines:

- BITV 1.0 (Level 2)
- Stanca Act

The tool offers various features to ensure that you're able to fix any known errors in web content and design:

- **The ability to visualize:** Use a browser to view accessibility tests with icons showing over the website image to indicate known, likely and potential issues
- **Check public or private applications:** Check public or private applications with basic authentication, CMS authentication or custom form authentication
- **Perform ongoing automatic testing/monitoring:** Use the schedule feature to ensure automatic testing and reporting on a monthly basis
- **Receive notifications when issues arise:** If an issue is found, you will receive email notifications to alert you of known, likely and potential issues

SiteImprove

SiteImprove allows you to ensure your website is inclusive and usable for every visitor that comes your way. This helps you adhere to various global accessibility guidelines, including WCAG 2.1, with features such as:

- **Accessibility diagnosis:** Use practical recommendations, your own criteria or the tool's scoring system to decide which issues you want to fix first

- **On-page and in-code highlights:** View issues with on-page and in-code highlights that make it simpler than ever to fix errors

- **Quality assurance/accessibility dashboards:** See how far you've come in terms of accessibility with customizable dashboards and automated reports

The tool lets you see every link, page and media file at a glance, locating issues that could impact your site's accessibility level.

Monsido

Monsido scans your website for any issues that may be stopping you from reaching ADA compliance with Section 508 and WCAG 2.1 standards. You can leverage this tool to:

- **Choose the compliance level you need:** Not all sectors have the same compliance requirements, so you're able to check for WCAG 2.1 or Section 508 accessibility issues

- **Leverage a built-in help center:** A help center is available via an icon for times when you're not quite sure how to fix an issue and would like some detailed information or instructions

- **View your accessibility status:** Compliance standards are organized by levels, such as A, AA and AAA, allowing you to fix low-level issues first and foremost before putting time into high-level issues

- **Reach out for training sessions:** Take advantage of training and one-on-one support options available to you and your team to ensure you're up-to-date on the latest best practices

The tool will scan your domain on a weekly basis to help ensure compliance at all times. This information can be found in the dashboard, wherein you can see a total count of issues and which pages they're found on.

Accessibility Testing Checklist

When it comes to accessibility testing, it might feel like there's a lot of work to do and, while that's true, a checklist can make the process a lot easier. Here's an accessibility testing checklist you can follow.

Text

- Can users enlarge the text if needed?
- Is there enough contrast between the text color and the background color?
- Does the website use bold, underlined and italics when needed?
- Are the navigation items labeled clearly in a way that's understandable?
- Is the font easy to read and, if not, can the user override fonts for text displays?

Video and Audio

- Do all video and audio clips fit with the content shown on the page?
- Are subtitles or transcripts available for every form of multimedia used?
- Can video and audio clips be stopped once the page loads?
- Are video and audio clips safe from dangers facing those sensitive to light and sound?
- Are users able to adjust audio or video controls as needed?

Color and Contrast

- Do background and foreground colors have enough contrast?
- Are there alternatives provided to color-coding used as a means of conveying information or indicating actions?

Images

- Do all images used have a purpose (to convey information)?
- Is the site loading slowly because of too many images?
- Do all images have alternate text to go with them?
- Are there low contrast/extremely bright colors that may be difficult to view?
- Are users able to stop images from flashing, rotating or moving?

Language

- Is the language clear and concise, without confusing terms?
- Are there links to explain words that might be advanced for an average vocabulary?

Forms

- Are text boxes within forms clearly labeled?
- Are text boxes organized properly and predictably (name, address, city, etc.)?
- Can forms be filled out using the keyboard to navigate from one box to the next?
- Are required fields indicated in a non-visual way?
- Are text error messages given for invalid fields?
- Does a success message appear to indicate submission?
- Are fieldsets used to group related controls?

Headings

- Are headings simple and descriptive so the meaning is clear to users?
- Is there a linked table of contents on pages with a lot of information?

For more information on what to look for when accessibility testing, check out this complete web accessibility testing checklist.

Conclusion

Web accessibility testing is fundamental to ensuring people with disabilities are able to use your website without barriers. Once your website is made accessible, you can rest assured knowing you're meeting your legal obligations while helping to improve inclusivity for all individuals, including the millions of people around the world with one or more forms of disability.

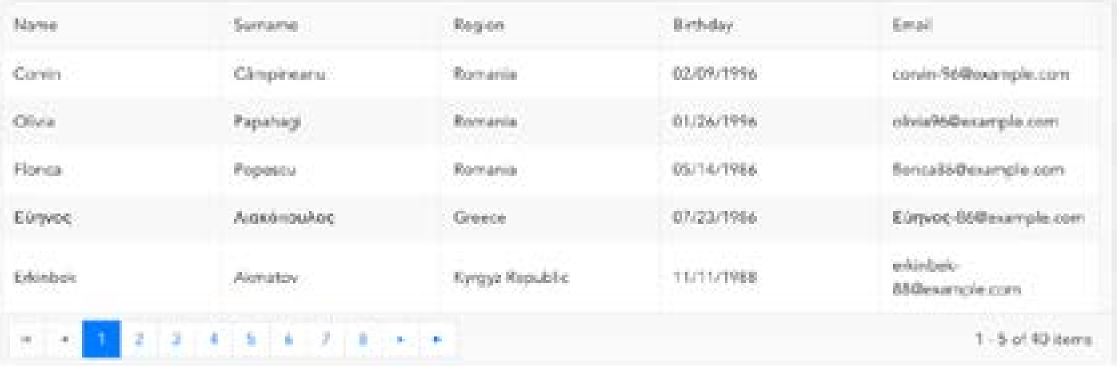
7. Example: Building Accessible Grids

In this section, we will look at how to implement a grid that is accessible using the Kendo UI Grid Components. Note that we are using a Kendo UI for jQuery component for this example. Kendo UI has native components for Angular, React and Vue as well.

Web accessibility means that people with disabilities can use the web, and how a grid is interpreted differs for sighted and non-sighted users. A sighted user can quickly understand the information because they have a visual of the entire grid. A non-sighted user interacts with a grid differently. Assistive technology, like a screen reader, will read the grid one cell at a time. Therefore, the grid must be designed so that contextual information is not lost. Up next, we will see how the grid achieves this and conforms to the Web Content Accessibility Guidelines (WCAG) 2.1.

Making Headers Screen Reader Accessible

The WCAG success criterion 1.3.1 states, “Information, structure and relationships conveyed through presentation can be programmatically determined or are available in text.” For a grid, the structure and relationships within the data are primarily identified by its headers. The headers are usually the first row or first column of a grid. Headers may also be distinguished from other cells with bold font or a shaded background. These visual cues cannot be interpreted by a non-sighted user. Furthermore, because the headers are always visible, a sighted user knows which data cell is associated with which header. The following is an example table.



Name	Surname	Region	Birthday	Email
Corvin	Câmpineanu	Romania	02/09/1996	corvin-96@example.com
Olivia	Papahagi	Romania	01/26/1996	olivia96@example.com
Florica	Popescu	Romania	05/14/1986	florica86@example.com
Εύηνος	Διακόπουλος	Greece	07/23/1986	Εύηνος-86@example.com
Erkinbek	Aimatov	Kyrgyz Republic	11/11/1988	erkinbek-88@example.com

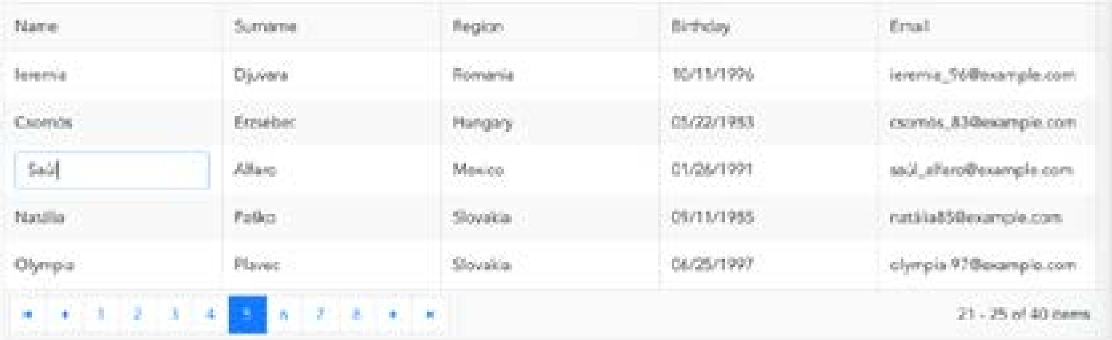
1 - 5 of 40 items

If the grid wasn’t designed to be accessible, the first row after the headers would be read as “Corvin, Câmpineanu, Romania, 02/09/1996, corvin-96@example.com.” In this example, one might be able to decipher what each piece of information means. But your data may not be as distinct. If you had a grid of financial data, each cell would be a number and it would be virtually impossible to determine what each number meant. For a non-sighted user, there needs to be a way to associate the header with a data cell.

For a table with a simple layout, using the `th` element to identify header cells is sufficient for a screen reader. However, for tables with a complex layout, the `scope` attribute should also be added to table headers. The table headers of the Kendo UI Grid have the `scope=”col”` attribute to associate column headers with the data. With these enhancements, a screen reader will now announce the name of the header with the data when the column changes. Using the previous example, the second row would be read as, “Name, Corvin; Surname; Câmpineanu; Region, Romania; Birthday, 02/09/1996; Email, corvin-96@example.com.”

Making Content Keyboard Accessible

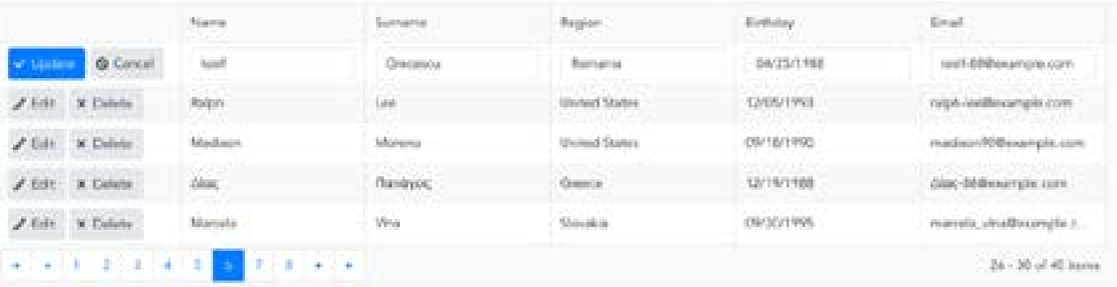
The next accessibility guideline we will review is making content keyboard accessible, which is in WCAG Guideline 2.1. There are users who have physical disabilities that prevent them from using a mouse so they rely on the keyboard 100% of the time. Even if a user does not have a physical disability, it can provide greater convenience to have the option to use a keyboard. A Kendo UI Grid can be navigated by keyboard by setting the `navigatable` attribute to `true`. This allows you to use the arrow keys to move throughout the grid. This also helps with editing the grid—if you have enabled in-cell editing, instead of clicking the cell to put it in edit mode, the user can press enter.



Name	Surname	Region	Birthday	Email
Ieremia	Djuvara	Romania	10/11/1996	ieremia_96@example.com
Csomós	Erzsébet	Hungary	05/22/1983	csomos_83@example.com
Saúl	Alfaro	Mexico	01/26/1991	saúl_alfaro@example.com
Natália	Faško	Slovakia	09/11/1985	natalia85@example.com
Olympia	Plavec	Slovakia	06/25/1997	olympia.97@example.com

21 - 25 of 40 items

In the following example, the editing mode for the grid is inline. But now the edit command can be activated by the keyboard.



	Name	Surname	Region	Birthday	Email
✓ Update ⊘ Cancel	Iosif	Oncescu	Romania	04/25/1988	iosif-88@example.com
✎ Edit ✕ Delete	Ralph	Lee	United States	12/05/1993	ralph-lee@example.com
✎ Edit ✕ Delete	Madison	Moreno	United States	09/18/1990	madison90@example.com
✎ Edit ✕ Delete	Ηλίας	Παπάγος	Greece	12/19/1988	ilias-88@example.com
✎ Edit ✕ Delete	Marcela	Vina	Slovakia	09/30/1995	marcela_vina@example.c...

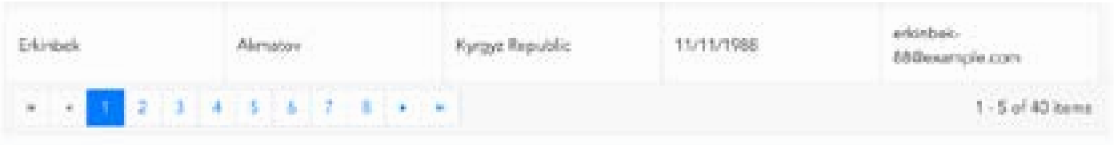
26 - 30 of 40 items

There is also a `navigate` event handler that can be used when `navigatable` is enabled. The event can be triggered with either the mouse or keyboard. Event handlers such as `onClick` that can only be triggered by a mouse limit accessibility.

Help Users Navigate

The last accessibility standard we will explore is WCAG Guideline 2.4, which says, "Provide ways to help users navigate, find content and determine where they are." The grid does this in several ways. When the grid is in edit mode, the input element that is in focus is highlighted.

When the selectable attribute is enabled, the selected row is given a contrasting background color. If `pageable` is enabled, a pager will be added to the footer of the table so you can see your location in the grid. By default, there will be a button for each page, with the button for the current page highlighted. The pager can also include an input element so the user can enter the page number to navigate to. Last, the number of data items on the page along with the total number is updated as you page through.



Erkinbek	Akmatov	Kyrgyz Republic	11/11/1988	erkinbek-88@example.com

1 - 5 of 40 items

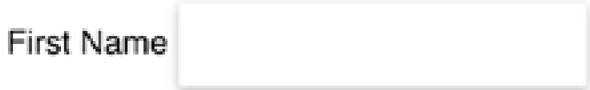
8. Example: Building Accessible Forms

We've covered the rules that you need to use when creating forms, and in this section, we will give an example of how you can make your existing forms accessible using a Kendo UI dropdown component. Note here that we are once again using a Kendo UI for jQuery component for this example. Kendo UI has native components for Angular, React and Vue as well.

Any user interacting with a form needs to be able to understand its purpose. For a sighted user, placing text near the form control is usually enough. But for someone using assistive technology, the name and value of a form element have to be programmatically determined. The solution to this is to use a label element with an input. This lets you associate a piece of text with a form control.

Using Labels with Forms

An input associated with a label will have text spoken by a screen reader when the input is in focus. Providing labels or instructions with form elements is one of the accessibility requirements set by WCAG 2.1. You can use a Kendo UI theme to style generic form controls by adding the appropriate class to the element. For example, you would add the class `k-textbox` to a text input. To make the form control accessible, you use the `for/id` method. This sets the `for` attribute of the label equal to the value of the input's id. The following is an example:



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset=”utf-8”>
    <title>Forms</title>
    <link rel=”stylesheet” href=”https://kendo.cdn.telerik.com/2019.1.220/styles/
kendo.common-material.min.css”>
    <link rel=”stylesheet” href=”https://kendo.cdn.telerik.com/2019.1.220/styles/
kendo.material.min.css”>
    <script src=”https://code.jquery.com/jquery-1.12.3.min.js”></script>
    <script src=”https://kendo.cdn.telerik.com/2019.1.220/js/kendo.all.min.js”></
script>
    <style>
    body { font-family: helvetica; }
    </style>
  </head>
  <body>
    <form>
    <label for=”fname”>First Name</label>
    <input id=”fname” class=”k-textbox” />
    </form>
  </body>
</html>
```

Using Placeholders with Forms

We usually recommend against using placeholder text—it tests poorly with users without a disability and can definitely cause problems for users with a disability. Placeholder text disappears when a user clicks in the field. This is a huge problem for users with cognitive/memory issues. We have on occasion used labels that were visible for screen readers only when a field’s purpose was visually obvious (e.g. search field with a search icon and search button). Static labels are always the best practice for accessibility and usability.

9. Final Thoughts

We started off first with an overview of accessibility, what it means to web developers and what laws apply. We covered different aspects of accessibility and how to implement features that support it. Finally, we gave a few examples of how to actually implement accessible components along with code examples using Kendo UI components. The examples used the Kendo UI jQuery components, although Kendo UI has native components for Angular, React and Vue as well.

On behalf of all the authors who contributed to this paper, we hope you have found it useful. Accessibility is one of those areas where we get an intersection of "things you have to do" and "things you can feel good about doing." By helping to gain a better understanding of how accessibility works, you will hopefully keep these techniques in mind when working on your next app. You'll wind up with a better app, generally increased usability overall and a satisfied user community, now that a larger portion can use your app.

We gave you a number of links throughout this paper for places you can go to get more information about accessibility. In the following section there are a few more places to go for extra info as well. Happy coding!

More Resources

Google's Introduction to Web Accessibility

Google offers an introductory guide (text version available here), as well as practice design labs and samples for those looking to learn web accessibility. There is no requirement of prerequisite knowledge, but it is helpful to have some background in web design. The course teaches what accessibility is, advanced accessibility techniques and more important information to create usable and accessible websites.

This course goes deeper into how accessible semantics can help with navigating a webpage, how screen readers work and how to find potential accessibility tweaks on an existing webpage.

The course is part of a nanodegree in Frontend Development and covers areas of accessibility such as the focus and the navigation without a mouse, the web semantics for screen readers and the general styling for accessible elements.

The course also shows how to use the developer tools built into the Chrome web browser to audit the accessibility of web pages and understand what changes can be made to improve the usability for all visitors.

International Association of Accessibility Professionals Certification (IAAP)

The IAAP offers a certification program that helps individuals add to their professional qualifications with proof of capabilities and/or knowledge of accessibility. The IAAP offers two types of certifications: a professional-level credential and a technical-level credential. You don't have to be a member of the IAAP to take any certification course, and prior experience is not necessary but helpful.

Microsoft's Training Teachers to Author Accessible Content

Microsoft offers an hour-long course separated into ten modules on web accessibility. A valid Microsoft account is necessary to complete the course on their website. Each module has course notes and a video provided to individuals taking the course. You will learn how to create and revise documents so everyone, including those with disabilities, can access them.

Mozilla’s Understanding Web Accessibility Guidelines

Mozilla compiled a great series of resources explaining exactly how to comply with WCAG 2.0 and 2.1. The course breaks down each of the four principles of accessibility (Perceivable, Operable, Understandable and Robust) and links to practical resources so that you don’t have to go hunting anywhere else.

The four sections, each covering one aspect of accessibility, are a comprehensive list of things to keep in mind when building quality websites. Mozilla’s compilation covers the basics of providing text alternatives to non-text content, continues on to instructions for making that text show up on the page in predictable ways, provides specific guidelines on webpage navigation and covers everything in between.

Many of the suggestions are a must-read for everyone building a webpage, for example:

- Making the text content readable and understandable
- Providing input assistance to help users avoid and correct mistakes

Don’t Make Me Think, Revisited: A Common Sense Approach to Web Usability (3rd Edition) – Steve Krug

This is one of the best books on usability available. While usability is about creating effective and clear sites, applications and software products, many of the usability concepts discussed in this book also make your website more accessible to people with disabilities and impairments—for example, those less sensitive to certain colors, or those with poor visual acuity or tunnel vision. This new edition includes a chapter on mobile usability.

In Chapter 1, with the same title as the book itself (“Don’t Make Me Think!”), Steve Krug points out that making every page seem obvious makes the website experience feel better to the users. “The sites that don’t make us think about unimportant things seem effortless. At the same time puzzling over things that don’t matter saps our energy and our time,” Krug argues.

Chapter 5, “Omit Needless Words,” becomes ever more important as new content gets added to the web every day. The author provides suggestions on making your writing on the web shorter and more effective.

Accessibility Wins

If you’re in need of a little inspiration, scroll through the examples over on Accessibility Wins. The blog features the websites that are making their functionality accessible to all. Know of any great examples of accessible websites? Submit your entry!

One of the things you can learn about on Accessibility Wins, for example, is Facebook's accessibility toolbar—a great example of navigation help on a large website that otherwise would be tough to get around without a mouse or a keyboard. Ideally, no website should be as complex as Facebook. But if yours is, this is an effective pattern to try out.

There is also a great explanation of the Google Maps keyboard-only navigation. Pro tip: press Shift+Tab twice to get right into the keyboard-only mode without having to tab through the entire webpage.

The A11Y Project

The A11Y Project has a great list of resources, including a library of accessible widgets and patterns, a checklist for an accessible website and a long list of screen readers, courses and software to check color contrast.

The accessible widgets and patterns are available for free from the Patterns page. There, you will find everything from buttons and forms to tables and even to a list of embeddable accessible video players. Many of the elements are written in vanilla JavaScript and don't require any extra libraries to work.

The Project also organizes accessibility-related conferences and meetups, and you can find the information on those in the events section. There are events of all kinds, from online webinars to in-person events around the globe.

Authors

Garenne Bigby

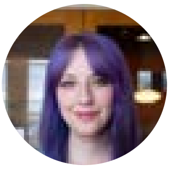
Garenne Bigby is the founder of DYNO Mapper, a Software as a Service (SaaS) that offers enterprise accessibility testing for websites and online applications. He is an Advisory Committee Representative at the World Wide Web Consortium (W3C) and a Certified Professional in Accessibility Core Competencies (CPACC).

Nada Rifki

Nada is a JavaScript developer who likes to play with UI components to create interfaces with great UX. She specializes in Vue.js and loves sharing anything and everything that could help her fellow frontend web developers. Nada also dabbles in digital marketing, dance and Chinese.

Chris Ward

Chris Ward explains cool tech to the world. He is a technical writer and blogger. He has crazy projects in progress and will speak to anyone who listens. You can talk to him! :)

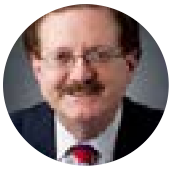


Bekah Rice

Bekah is senior UX designer/developer and accessibility expert at user experience strategy firm truematter. Her dual skillsets in visual design and frontend development help her create simple, usable designs that best serve the needs of real people online. She has worked with Fortune 500 clients on end-to-end UX and development projects. You can read more accessibility and usability articles by Bekah at blog.truematter.com/author/bekah.

Baley Lewis

As director of content strategy at user experience consulting firm truematter, Bailey oversees the content and writing initiatives for all UX projects. Her work is informed by researching and testing digital products with real users. She works with regional, national and Fortune 500 clients on UX and content strategy projects, from definition through implementation. You can read more accessibility and usability articles by Bailey at blog.truematter.com/author/bailey.



John Willoughby

John has held various software development and product marketing roles at both hardware and software companies, and is currently the Product Marketing Manager for Kendo UI. He has a Bachelor's in Electrical Engineering (Computer Design) and is currently working on his Master's in Computer Science. When not actually sitting in front of a monitor, he enjoys playing guitar.

Alberta Williams

Alberta is a software developer and writer from New Orleans. Learn more about Alberta at github.com/albertaw.

Get a head start creating accessible web apps with components from the Kendo UI libraries. Find out more.

About Progress

Progress (NASDAQ: PRGS) offers the leading platform for developing and deploying strategic business applications. We enable customers and partners to deliver modern, high-impact digital experiences with a fraction of the effort, time and cost. Progress offers powerful tools for easily building adaptive user experiences across any type of device or touchpoint, award-winning machine learning that enables cognitive capabilities to be a part of any application, the flexibility of a serverless cloud to deploy modern apps, business rules, web content management, plus leading data connectivity technology. Over 1,700 independent software vendors, 100,000 enterprise customers, and two million developers rely on Progress to power their applications. Learn about Progress at www.progress.com or +1-800-477-6473.

2019 | RITM0043896

Progress®