

# A Product Leader's Guide To Accessibility & AODA

---

AN  
**ENGINESS**  
BUSINESS GUIDE

## Table of Contents

Table of Contents	1
Introduction	2
About us	4
01 Understanding accessibility regulation in Ontario	5
02 The business case for compliance	15
03 Project delivery	26
04 Managing costs	43
05 Ongoing maintenance	58
06 Common risks and pitfalls	63
07 Final thoughts	75

# Introduction

From a morals and ethics perspective, there's no doubt that making a continuous and conscientious effort to design and deliver accessibility compliant digital products and web applications is a good thing. As we'll see, it not only brings the value of your digital service to more people, but it genuinely creates a better experience for those you serve, regardless of their level of ability.

For business owners and project sponsors however, it's often difficult to know the difference between ADA, AODA, and WCAG 2.0 and what standard they are required to, or desire to, meet, let alone how to properly achieve that standard during planning, design, development, testing and post-launch periods.

We've worked on dozens of accessibility projects over the years, and have helped nonprofits, associations, businesses, and regulated industries navigate the tricky waters of accessibility compliance.

Over the years, we've picked up plenty of tricks to help accelerate projects and deliver a positive outcome to digital leaders, organization leaders, and end-users. And now we've condensed those years of learning in this ebook, jammed with actionable advice to help you execute your own project where accessibility is a requirement.

In many organizations' digital strategy roadmap, accessibility is an important priority. In this ebook, we're going to deep dive into this single line item on a digital project, and cover the entire "accessibility-in-mind" lifecycle within your project.

**TIP:** Want to learn more about digital strategy roadmapping? [Visit our Insights](#) to read our related posts on the topic.

Simone Abel, Director of Digital Strategy at Enginess, shares her expertise on the process of producing accessibility compliant digital experiences, as it relates to achieving digital transformation. In this ebook, you'll learn:

1. What the AODA regulation is, and where you fit into the compliance framework if you are an Ontario company doing business online;
2. What WCAG 2.0 Level A, AA, AAA standards are, what they mean to you, and the role they play in governing AODA and ADA compliance;
3. The different conformance levels to consider in order to meet the needs of your users;
4. How's and the why's of building a business case for accessibility;
5. What you need to do to incorporate accessible design into your way of working, and how to design, develop, and deploy digital products in an accessible way;
6. How to derive strategic design and technology direction to build and price your project more accurately;
7. Common pitfalls, and how to overcome them.

We're going to walk through each one of these lessons in detail, and by the end of the ebook, we hope you'll have a good understanding of why you need to build with accessibility criteria in mind, how to go about it, what it costs, and how to make your project a success.

## About us

Enginess is a Toronto-based digital consultancy. We empower businesses to get the most out of technology.

We shape strategies for business processes and deliver solutions that enhance customers' experience, improve efficiencies, generate new market opportunities, and redefine value creation for many different kinds of organizations.

We offer services that include:

- Technology Procurement
- Digital Business Strategy
- Revenue Growth Strategies
- Website, eCommerce and Custom Software Development
- Experience Design

Over the past 18 years, we've built a solid reputation based on our ability to consistently deliver high-quality customer service and handle complex digital strategy and development assignments.

1.416.901.6151  
info@enginess.io  
enginess.io

 @enginessio

 @enginessio

 @enginess

# 01 Understanding accessibility regulation in Ontario

If you're a digital leader doing business out of Ontario Canada and spearheading a web project, you've most definitely heard of the [Accessibility for Ontarians with Disabilities Act, or AODA](#). First signed in 2005, the AODA is :

*"Ontario's accessibility laws and standards, our action plan and progress in becoming accessible for people with disabilities by 2025."*

The regulation provides detailed guidelines for accessibility compliance in Ontario, as well as the detailed schedule of what compliance standards need to be met, and by whom. The act covers everything from physical construction of spaces (e.g. who needs wheelchair ramps and when) to digital spaces (e.g. providing adequate colour contrast to improve readability of a website). We're interested in the digital compliance component of the AODA.

For those who are new to it, the AODA is at best difficult to navigate and at worse a bamboozling mess of technical terms, legal rules, and exceptions. Furthermore, the AODA has been designed to be rolled out gradually over 20 years with different deadlines for different organization types. The level of compliance you're expected to achieve and when you need to achieve it is different depending on your

organization. Where to begin? Here are some key questions you need to ask if you're a business or nonprofit before you even start your digital project.



## 1. AODA jurisdiction

AODA is a provincial regulation that only governs organizations doing business out of Ontario, Canada. If you're a digital product owner based in Mexico, then you don't need to legally comply with AODA. However, as we'll see shortly, AODA offers significant **global** benefits for website owners beyond complying with regulatory requirements, so we recommend everyone works towards these standards regardless of geography. At time of writing, Canada, USA, Australia, New Zealand, EU, and parts of South America and Asia had introduced their own web accessibility laws or adopted WCAG 2.0 standards. Within Canada, Nova Scotia and Manitoba have enacted accessibility legislation, with British Columbia in the process of proposing new legislation. [Research](#)

[your country](#) and specific province or territory to understand what your legal requirements are.

## 2. Business definition

Depending on the type of business organization you are, the requirements and timelines differ. There are six different categories of organization, and diving into the specifics of each one, their deadlines, and the level of compliance required for each would be a whole book in its own right. So for now, we're just going to focus on private, for-profit businesses, enterprises and nonprofits. The question is: who counts as a business vs. nonprofit?

The easiest way to answer this is to look at the other categories, and rule out your membership in one of those groups:

- Public sector organizations
- Municipalities
- Educational institutions, including boards, colleges, universities, and schools
- Textbook producers
- Library boards

If you're not on this list, odds are you're a business or nonprofit according to the AODA guidelines.

## 3. Number of employees (company size)

Finally, how many employees do you have? This includes anyone you pay a wage or salary to, including full-time, part-time, seasonal, and contract workers. You don't need to count volunteers, independent contractors, or employees outside Ontario.

There are three bands of employee count with the AODA framework:

- 1-19 has one set of deadlines



- 20-49 has another
- 50+ has a third.

By 2025 all three should arrive at roughly the same place in terms of web accessibility compliance, but they'll get there at different timelines according to the type of organization and size.

## 4. What about ADA?

Closely related to AODA is the [Americans with Disabilities Act, or ADA](#). The ADA regulates American organizations in meeting and serving those with disabilities, and includes language in Title III that requires websites to be made accessible.

Much like the AODA regulation, ADA compliance is designed in part to make digital properties usable for everyone. [ADA includes design standards for web properties that were written in 2010](#), but as its been stress-tested in courts, the governing standard that's emerged is the WCAG 2.0.

With that in mind, plus the authority of the W3C, we recommend that all organizations design to WCAG standards to comply with both ADA and AODA regulations.

Now that we have a better understanding of *who is* supposed to pay attention to AODA, let's start to take a look at *what* that means for production teams.

## WCAG 2.0

The AODA web compliance requirements are based on the [Web Content Accessibility Guidelines \(WCAG\) 2.0](#), developed and maintained by the [World Wide Web Consortium, or W3C](#). This is fairly standard across web accessibility regulatory frameworks, including the AODA and ADA. So before we get into what different

organizations and sizes are required to have completed by when, it's worth doing a review of what WCAG is and why everyone uses it as the central governing source.

Broadly speaking, WCAG is there to help make websites more accessible to those less able, particularly those with visual or hearing impairments. Out of that objective has come an incredibly detailed outline for how to build a website for maximum compatibility, covering everything from metadata best practices to colour contrast guidelines.

The WCAG (and associated frameworks provided by W3C like [WAI-ARIA](#)) is the true heart of web accessibility, and regardless of whether you're designing for a regulatory framework or just building a site for fun, it's definitely worth using the WCAG to help guide your decision-making.

WCAG is divided into four principles:

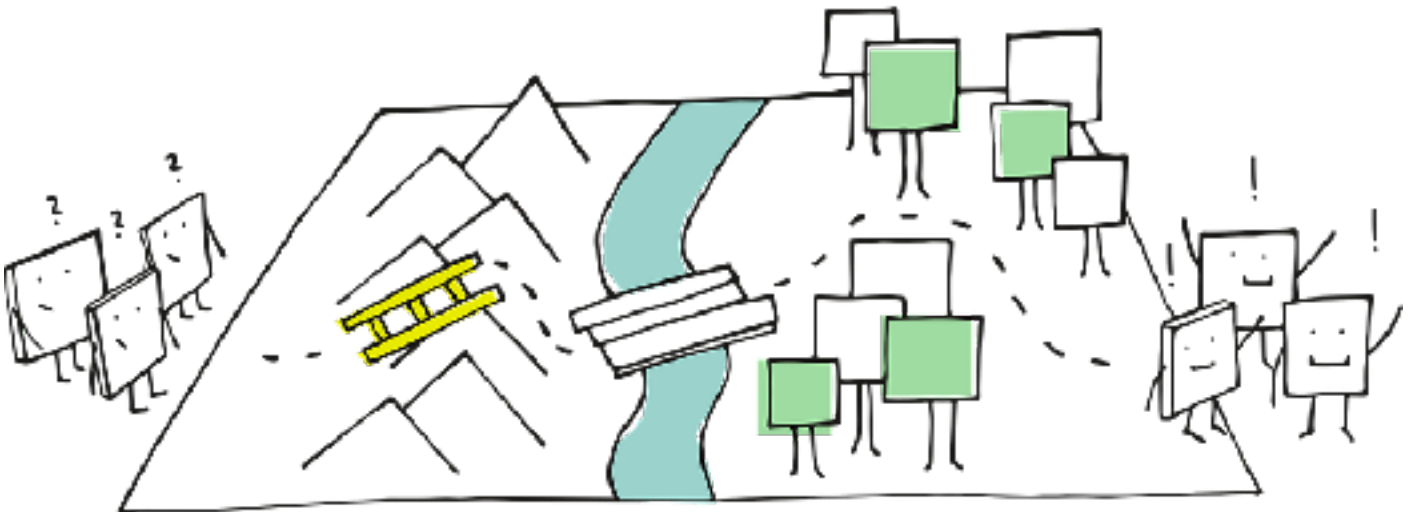
1. **Perceivable:** Are you presenting information in a way users can perceive it? Can users who are vision or hearing-impaired, or working with poor internet connections still use your website?
2. **Operable:** Can people navigate your website with a keyboard or other basic input method (e.g. a mouse)? Can users read and consume the content on your site without being rushed or the pages changing? Can users navigate your website, and can they use your site even if they're at risk of seizures?
3. **Understandable:** Can your website be easily read and comprehended by most audiences? Can people generally find what they're looking for and avoid / correct mistakes easily?
4. **Robust:** Does your website work with lots of different browsers, devices, and assistive technologies, including backwards compatibility with older tech?

Because WCAG 2.0 is so detailed, they've built out three different tiers of accessibility:

1. Level A
2. Level AA
3. Level AAA

While some would love for the online world to be strictly AAA compliant, the reality is that it's an incredibly difficult designation to meet. For most businesses and nonprofits, AA is a great level of accessibility to aim for that achieves most of the benefits of accessible design without driving costs and complexity through the roof. This is why AODA only requires AA compliance for businesses and nonprofits above 50 employees.

The last thing to understand about WCAG 2.0 is that it is not the easiest documentation to read. We often find that while client core teams may understand their legal obligations and/or some part of the guidelines, they don't know how to design, test or validate for compliance. Ensuring the talent and skill set exists in-house isn't often feasible or viable, so hiring an external expert at some point in the process is necessary.



## What WCAG 2.0 Level A and Level AA mean

If you're building towards WCAG 2.0 Level AA, what does that actually mean?

Here's a checklist of what's required. There's a ton of resources that get into detailed specifics on tactics and implementation, but we're going to keep it high level here.

### Level A

- Provide text alternatives to non-text media: images, videos, audio, etc...
- Provide alternatives to time-based media (videos or audio that automatically start playing)
- Content must be adaptable / able to be stripped down without losing meaning
- Colour contrast must be high enough, fonts large enough to easily read
- If audio auto-plays, users must be able to stop it
- Full functionality must be possible from the keyboard
- Content is on the screen for long enough to read
- Content can't flash or induce seizures
- Content must be structured, titled, in a focused order, and links must be clearly labelled and go where they say
- The default language should be able to be programmatically decided
- The user experience must be predictable, with actions have predictable outcomes
- Errors have to be identifiable and correctable
- Websites have to be compatible with previous and future generations of assistive technologies

## Level AA

Level AA requires everything in level A as well as:

- Meet minimum contrast requirements
- Text must be resizable
- Text must be an editable font so that users can manipulate it as needed
- There must be multiple paths to different pieces of content
- Headings are clear and describe the page they're on
- The action that's possible to take on every page is clear
- The language used on your website has to be automatically detected
- Every component on your site must do the same thing in every instance it appears
- Legal and financial transactions and user-generated data storage must be controllable, reversible, checked, and confirmed
- Errors must be auto-detected, and solutions suggested

**NOTE:** NOTE: At time of writing, W3C introduced a new extension to WCAG 2.0 with [WCAG 2.1](#):

- Content that conforms to WCAG 2.1 also conforms to WCAG 2.0
- The publication of WCAG 2.1 does not deprecate or supersede WCAG 2.0
- While WCAG 2.0 remains a W3C Recommendation, the W3C advises the use of WCAG 2.1 to maximize future applicability of accessibility efforts
- The W3C also encourages use of the most current version of WCAG when developing or updating Web accessibility policies
- The WCAG 2.1 [Quick Reference](#) provides ways to view success criteria grouped by conformance level, along with many other filter and sort options.

## AODA web compliance requirements

Now that we understand AODA regulations and WCAG 2.0 standards, let's look at specific requirements for the three tiers of business and nonprofit organizations.

### Accessibility requirements for businesses and nonprofits 1-49

For businesses and nonprofits with under 50 employees, **there are no web accessibility requirements enshrined in AODA** (section 14 of the Act). However, it's still best practice to follow the path, if not the timeline, of larger organizations.

### Accessibility requirements for businesses and nonprofits 50+

Organizations with 50+ people do have accessibility requirements. By now, organizations with more than 50 people should have:

- Designed an accessibility plan and posted it onto an accessible website (2014)  
Made any new / significantly updated website accessible to WCAG 2.0 Level A (2014)

What's more, by **Jan 1, 2021**, businesses and nonprofits should have made their websites accessible to WCAG 2.0 Level AA (except for the regulations around live captions and audio descriptions).



## Chapter summary

- AODA is the Accessibility for Ontarian's with Disabilities Act (AODA) and governs businesses and nonprofits in Ontario.
- The web accessibility sections of the Act are based on the Web Content Accessibility Guidelines 2.0 — similar to other accessibility acts around the world (e.g. ADA).
- AODA has different requirements and timeframes depending on what sort of organization you are if you're in Ontario, and how many employees you have.
- WCAG 2.0 has two relevant levels of accessibility to meet: Level A, which covers most accessibility basics, and level AA, which goes into more detail and builds on the framework provided in level A.
- By now, all websites should be accessible to level A. By Jan 1 2021, all businesses and nonprofits with 50+ employees are required to be accessible to level AA.

## 02 The business case for compliance

In the previous chapter, we covered what is required to meet AODA compliance standards, where the framework comes from, who needs to meet compliance standards, and when the deadlines are. In this chapter, we're going to focus more on the business side of things — what project sponsors and product owners need to do internally to provision adequate budget and resources for the production work.

### Why build a business case for AODA?

For some, the threat of running afoul of AODA or ADA enforcement teams, and the substantial fines they can levy, is enough to facilitate action on behalf of the organization. For others, it is a values-based decision based on best practices and transforming future-forward.

We've found from working with dozens of organizations on their digital projects that building a firm business case for baked-in accessibility compliance generally leads to better buy-in, better project planning and thus, better outcomes for everyone. It's usually a must-have requirement on most product owner's [digital strategy roadmap](#). We're going to break down the two sides of the compliance



business case — risk and reward — and provide a framework for how you can build your own.

## The risk side of the business case

Building a business case for accessibility compliance is mainly about understanding and quantifying the risk and/or opportunity loss.

### The financial risk of court cases and fines (liability)

First, what level of exposure do you have if you fail to meet compliance standards? Second, what's the total cost of failing to meet these standards? Are there spin-off costs and risks associated with a fine or violation?

The first part of this question is relatively easy to answer. AODA fines violators along two axis — the severity of the infraction, and the number of previous infractions an organization has had. For example, if you're an organization and it's your first infraction, and it's a minor one (e.g. administrative) then you'll be fined \$500 if you're a corporation, or \$200 if you're unincorporated / an individual.

If you had a major violation, and had a history of past violations, a corporation could be fined \$15,000 per day and can be fined up to \$100,000 per day.

There's some opportunity cost (which we'll cover in a second) associated with failed compliance, but generally speaking, there hasn't been significant fallout from a brand / PR perspective for any organization in Ontario...yet. The same cannot be said of the litigious environment in the US, where ADA fines are becoming big business. Scandinavian Airline System was fined \$200,000 in 2018 by the US Department of Transportation for violating accessibility laws not dissimilar to AODA regulation. At the same time, [The Toronto Star](#) reported that "thousands of businesses" are behind their accessibility requirements.

In short, it's only a matter of time before the government - and the public - begins prosecuting AODA violators with the full power the Act gives them.



## The reward side of the business case

At the other end of the AODA compliance business case isn't fear of loss, but potential gain to drive adoption from new users.

Because of the way the W3C has designed its WCAG 2.0 standards, there's significant value in following the framework. In a lot of ways, W3C has simply codified the principles of user experience then overlaid it with technical requirements to ensure that the user experience is consistent for users everywhere.

And if we've learned anything, it's that inclusive and delightful user experience pays the bills.

On top of a general improvement in user experience, the business case for accessibility can be broken down into three parts: innovation, market size, and brand equity.

Let's dig into these a little more.

## Innovation

Building a compliant website is a way to drive innovation at your company. Designing in an accessible way:

- Forces creative solutions, and forces designers to think of clever, non-standard ways to solve everyday problems. These solutions can go on to become the better solution overall, and eventually become mainstream. For example, the telephone, text to speech, email, and voice controls were all originally designed for those with disabilities, only to find broader application later ([source](#)).
- Designing digital products that might be consumed in a non-digital / less screen-intensive way forces designers to think not as a user sitting at a desk focusing on using the product, but rather as someone who maybe can't see the screen or doesn't have a full keyboard — conditions that are all too common.

## Market size

15% of the world's population has at least one disability for a total market worth of \$7 trillion. And as life expectancy continues to climb, the value of the market will increase. If organizations can design their services to meet some of these consumers where they are, they can capture significant revenues.



## Brand

Finally, brand. Closely tied to market share, building a website that caters to all users can build powerful brand equity. Especially for larger organizations with a dedicated corporate social responsibility charter (CSR), accessibility can be a powerful lever that organizations can pull. Particularly for consumer brands focused on being the “product for everyone” accessibility can be a superb tie-in with an overall brand strategy. Barclay’s, for instance, aims to be the “bank for everyone” — which makes accessibility a natural fit ([source](#)).

## Use cases

The last piece of the business case is understanding the use cases of those you're building for. Because remember: when we consider accessibility, we're not only designing for those with a disability, but also for *those who are disabled in the moment*. For example, someone who can't see their phone screen because of the glare from the sun is temporarily blind (from a design perspective).

In all my years of experience working with organizations, we've found a few use cases that come up again and again. There are *\*many more\** use cases we can talk about, of course. To help illustrate, here are two personas you could/should be keeping in mind as you approach accessibility and content for each persona. Consider how these users could improve your business if your website is accessible!





## The elderly

The elderly are a growing segment of the population who all too often are ignored when it comes to design.

However, this not only does a disservice to a big chunk of users, but it's also bad business. The elderly tend to switch products less often, be more loyal, and generate more revenue than younger customers. So they're a demographic worth designing for. At the same time, because elderly people often have more demanding accessibility requirements, if you design for them, you over-design for other core demographics, lifting the calibre of your user experience for everyone. Here's how:

- Make videos, PDFs, and images easier to read, with larger fonts
- Provide closed captions on videos and audio clips
- Avoid blue for important user interfaces (ageing causes problems in the eyes that make blue especially difficult)
- Allow fonts to be resized as needed
- Minimize front-end animations and effects
- Focus on your content readability and usability

## Low / medium / high reading comprehension level

Not everyone has the same level of reading comprehension, and odds are, it's lower than you think. 43% of Americans have a reading level at or below the 6th grade ([source](#)). At the same time, consider the overlay of how much attention a reader is paying to your content at the time they happen to be viewing it. Maybe they read fully-focused at a university level... but when they're glancing through your site with a million other things on their mind, they only read at an 8th-grade level.

Here's what you can do to improve your readability:

- Reduce words, word complexity, word uniqueness, sentence complexity, and sentence length
- Break up content into short digestible, easy to scan pieces (a principle called [“chunking”](#))
- Keep language clean and simple...consider those in your audience who aren't native speakers of the language of your website (English)
- Offer simple(r) visuals, videos, audio and images to communicate in different ways

## Conformance level

Conformance level is another core consideration when you're evaluating your accessibility project. When writing your [website accessibility legal statement](#), it's an important criteria to mention in terms of what a user may expect.

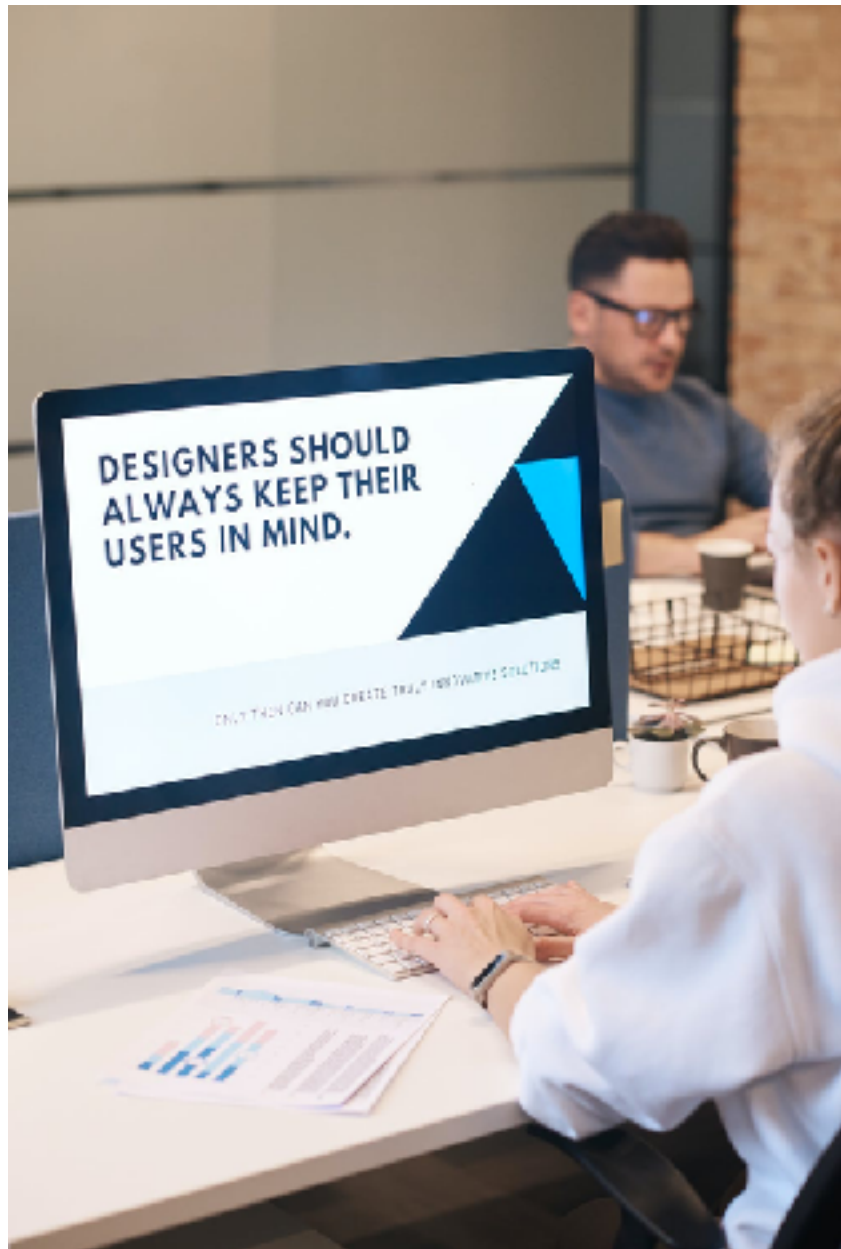
**Conformance level** is **essentially the degree to which you're complying** with a specific piece of the WCAG. There are three levels of conformance: non, partial, and full.

- **Non-conformance** is the lowest. It means that the content / interaction you're producing and hosting on your site does not meet the relevant accessibility standards.
- **Partial conformance** is when some aspects of your content or interaction meet the accessibility standards, but there are some parts that do not.
- **Full conformance** is when your content and experience meets your target accessibility standard without any exception.

Non-conformance and full conformance are fairly straight forward - in or out. However, most accessibility efforts are going to land somewhere in/around partial

conformance, perhaps working towards full conformance, if that's the stated intention of the business.

One example of partial conformance is when you rely on third-party content on your site. For example, say you have a blog. You might be using a comment tool like [Disqus](#) to power your comments, and you notice that the field labels on the comments forms disappear when you interact with them. Rather than completely changing your comments provider or, building your own functionality, you might simply acknowledge that some of your web pages partially conform because you use third party plugins that you can't control. E-commerce platforms and utilities are also pretty tricky.



## The difference between conformance and A, AA, and AAA compliance

It might seem redundant, but *conformance* levels and *compliance* levels are very different.



*Conformance* is about the degree to which you're meeting the requirements outlined by the W3C. *Compliance* is a rung up the compliance ladder, and is more about whether or not you've met the standard for that specific letter grade (A, AA, AAA).

Of course, there's an interaction between these two. For instance, if you have partial conformance on every AAA requirement, are you AAA compliant? This is almost a philosophical question, and it's easy to end up deep in the weeds on exactly how compliant you are. That's why we recommend ample user testing throughout the process, to keep everyone grounded in why we're doing this: it's not to meet compliance standards. Rather, we're doing this to make our website easier to use for all kinds of people. If that's your North Star, then meeting and exceeding compliance requirements from a legal perspective will naturally follow.

## Chapter summary

- A business case is an essential part of any compliance project. It helps deliver buy-in and ultimately leads to a better project.
- Part one of any web compliance project is about risk prevention and management. Basically, how much does it cost, and what's the risk of being fined or lose business if you neglect your obligations?
- Part two of web compliance is about increasing profits rather than avoiding loss through better innovation, increased market size, and improved brand value.
- Just two of a myriad of plausible use cases to consider: the elderly, a huge market potential and a group with specific needs and requirements; and those with low reading competency or English language comprehension.

## 03 Project delivery

Now that we're beyond business requirements, we're going to go deep into the product / software development lifecycle (SDLC), and where accessibility compliance fits in. In this chapter, we'll cover:

- User experience design
- Visual/graphic design
- Compliance-friendly QA
- User acceptance testing (UAT)

By the end of this chapter, you'll know what needs to be done throughout the production lifecycle to finish with a website that meets your stated compliance, accessibility, and usability goals.



## UX considerations

During the user experience planning stages of your project, it's critical to consider SDLC. And there are a few reasons for this:

- As we've mentioned, designing for accessibility generally leads to better designs
- UX is usually one of the first teams involved. The earlier accessibility and compliance can be brought into the project, the better the outcome
- UX is fundamentally about serving users, no matter what their level of ability
- Designing for accessible websites is, in many ways, fundamental to UX design
- UX is a natural starting point for accessible conversations

Now, of course, user experience is a broad field. Depending on the designer and the organization, UX can (and should!) be involved in every aspect of the product development lifecycle, including user interface (UI) design, branding, acquiring customers, on-boarding, and more. But accessibility is especially important. Peter Morville even put accessibility on his [UX honeycomb back in 2004](#).

So UX often becomes the accessibility champion on the web development team because it's so critical to how the rest of the job gets done, influencing every team to make better, more accessible decisions.

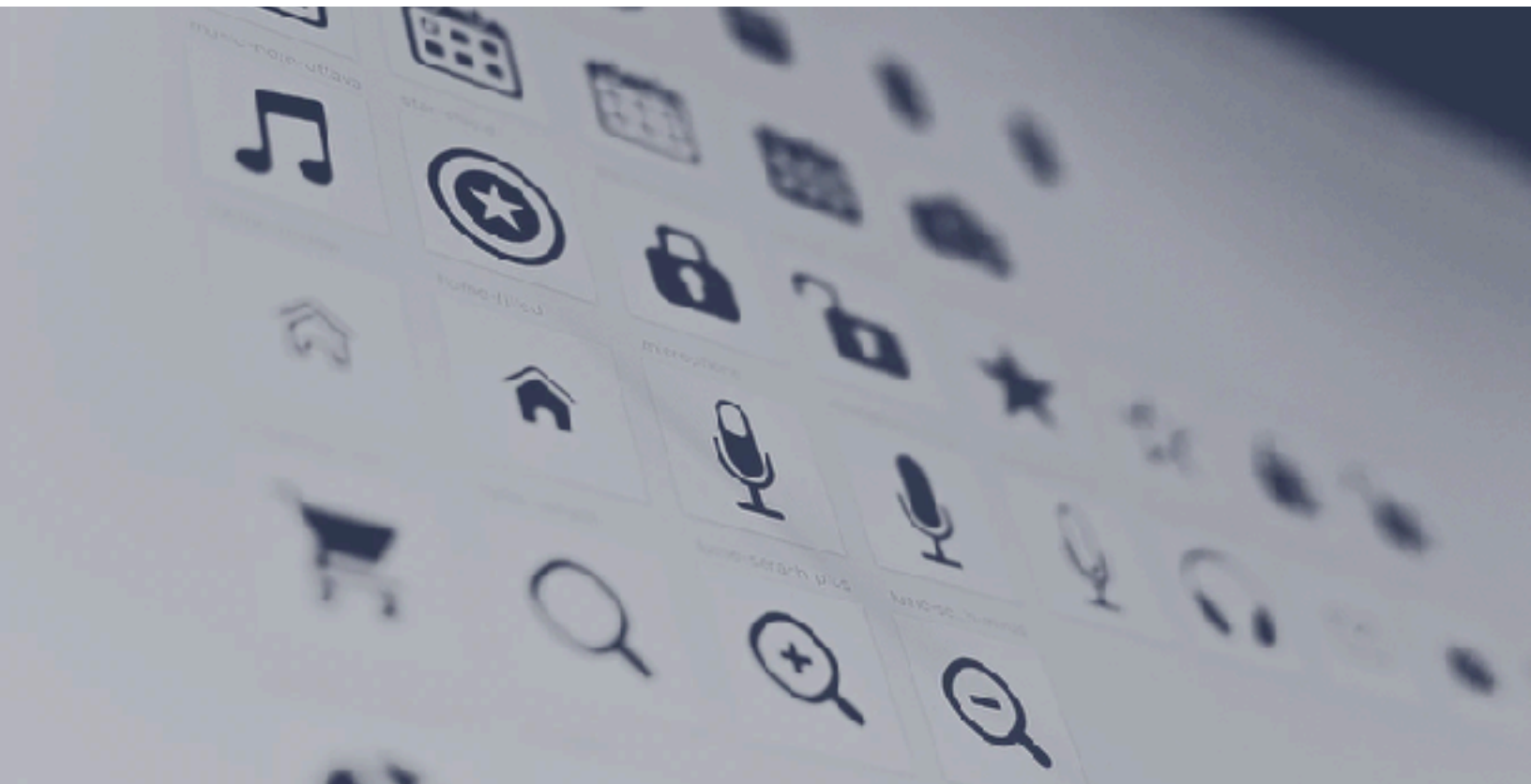


However, there are a few areas in particular where UX design can drive change:

1. Patterns and interactions
2. Technology and compatibility

## Patterns and interactions

In the context of usability and accessibility, patterns are recurring elements used throughout an application or website, particularly those elements created with rich languages like JavaScript. They almost always involve a heavy element of interaction design, because they're usually the pieces of your site that users actually interact with.



[Design patterns and widgets include](#) things like breadcrumbs, buttons, accordions (show / hide functionality), carousels, menus, radio buttons, checkboxes, and more.

UX designers should be looking out for:

- **Helping the user understand what they are interacting with:** Are they hovering over a button or scrolling through a feed to realize what the purpose or intent of the content is? Users should understand what element they're interacting with at all times, regardless of their technology or capability.
- **Building for screen readers and keyboards:** Whenever you're designing, design with both screen readers and keyboard-only users in mind. With those two use cases combined, you'll cover most of the use cases around accessibility.
- **Minimal to none animations, transitions and other front-end effects:** Moving elements on the screen decrease predictability. Think about the purpose of your content and your audience. Information sites, retail sites, business applications, software... need to be simple and easy to use.
- **Mobile vs. desktop presentation and usability:** Design for responsiveness and straightforward top-bottom, left-right stack and unpack of body content elements. Header and footer components simplified. Information architecture beyond three levels deep can be hard to access via mobile menus.

The great value that UX designers can offer early into the product development lifecycle when it comes to patterns and systems is *consistency*. Is every page being designed with the same elements and components in the same way? Does every interaction operate the same way, and do those interactions do what users expect them to? Can every page meet compliance standards from a design and

interaction perspective? Your UX designer is often the best positioned resource to keep every pattern consistent across a project and set the standard for compliance and conformance level early in the project.

This, in part, is contributing to the explosion of design systems and front-end design frameworks available today. It's also more expensive to design and code for "one-offs". To set up well for subsequent steps in the development cycle, your design systems need to be designed with AODA in mind.

## Technology & compatibility

[Bill Gregory, co-organizer of #A11yTo, tweeted in 2015:](#)

*"When UX doesn't consider ALL users, shouldn't it be known as "SOME User Experience" or... SUX?"*

And this is a good point (not to mention funny). The point he's making is that UX that is good for some users and not others isn't a good user experience at all. Rather, it's the UXer's job to make it a positive experience for everyone.

And while patterns, visual design considerations, and testing all play a role during the SDLC, it's all moot if you don't "design for everyone". But is that realistic?

This where assistive technology comes in.

Screen readers, keyboard-only users, text readers, speech-to-text, and alternative input methods like eye trackers, and head pointers are all scenarios that a UX designer needs to consider.

More practically, this means a few things.

First, meeting accessibility standards is a lot more important than you think. It is easy to bend the accessibility or lower the standard you are meeting for a design or

development consideration. But very quickly, you end up at SUX instead of UX. That's part of the reason so many organizations get help from accessibility experts – it's just more practical to keep projects on track.

Second, make sure your information architecture makes sense. **Accessibility to** information is just as important as **accessibility of** information. The problem that often plagues accessibility proponents in UX is that while the patterns and interactions are accessible, the underlying architecture or findability of the site is unintuitive, and information on each page isn't laid out in a hierarchical way (either because it's not tagged correctly, coded semantically, or it's simply not there).

Even if the patterns and interactions are completed beautifully and are fully compliant with WCAG standards, if the page and site organization is confusing, then those with assistive technology will struggle, as well as everyone else!





## Visual design considerations

Graphic / brand / creative designers have a significant role to play in achieving accessibility standards. That's because they have the biggest impact on what the end-user sees, they're often charged with enforcing or defining brand guidelines and building out the visual components needed for the patterns and interactions UX designers have outlined.

Here are a few things you should consider.

### Style patterns and interactions

WCAG 2.0 has clear standards on colour, patterns, and interactions that visual designers will need to adhere to. Fortunately, of all the accessibility requirements, this is probably the one that's best-documented and has the most rules of thumb to follow, not to mention good lists of tools readily available online.

Rather than write a laundry list requirements, we thought we'd frame this as a series of questions you should ask yourself before you release your design into the wild:

1. Is there enough contrast between every colour?
2. Is my typography clear, large, and spaced out enough to read easily, even for those visually impaired?
3. Does my design follow a linear path that mirrors both the importance of and content and mirror the placement of the headers?
4. When colour signifies meaning, is there another signifier as well (pattern on top of colour, etc... )
5. Do your visual elements have useful alt text that helps users?

6. Do all your interactive elements have 3 stages and do those stages map to screen readers?
7. Can someone who can't see your design still interact with it, without losing core functionality?
8. Are your focus indicators clear? Can a keyboard-only user understand and interact with them?
9. Do your icons and information images have context and labels?

The last point is, just like UX design, is consistency. Your visual design — and the baked-in alternative signifiers — must be consistent across your site. This not only leads to a better product but if you can keep your graphic assets effectively, it will actually streamline your design processes because you're reusing, rather than recreating, the assets you need.

As usual, following these sorts of design principles will make your website better in the long run, not only for users with challenges but for everyone.

## QA considerations

No website is finished without significant quality assurance (QA). The QA phase of a waterfall project comes after all development is finished, sometimes prior to content entry, and is an internal-facing project team activity. If you're working agile, you'll do QA on each release in your roadmap queue. The QA phase is primarily focused on functional and performance testing.

And it's where you can head off AODA compliance issues before hitting the end of a project. In a lot of ways, a good project is made or broken during QA testing. This is even more true for accessibility compliance.

On top of the obvious checks that need to happen during QA for accessibility requirements like checking all the considerations we just outlined, there are more technical requirements that must be adhered to in order to meet compliance standards.

Again, rather than getting into the specifics of the regulatory requirements, we're going to present a series of questions that you can use as a checklist with your team to achieve the level of compliance that you want.



## Front-end development - code evaluation & remediation

First, front end code. The best starting point is the WAI-ARIA guidelines. Those are used to govern how a website is developed as you move beyond the most basic HTML tags. Furthermore, common frameworks like Angular, React, and Vue have toolkits to help you as you're coding, so building accessibility from the ground up is getting easier.

But that's only half the battle. The other half is testing it.

A core requirement, and a theme throughout this ebook, is the idea of designing for screen readers. And there's a good reason for this: screen readers are to those with disabilities as screens are to everyone else. That is, ubiquitous, endlessly useful, and the main interface between us and the computer we're trying to use.

The trick, though, is that screen readers can't see all the beautiful design work that you've poured your heart into. Rather, they can rely on correct heading placement, correct mark-up, HTML tags to designate different elements, and alt tags to see what they can't see.

And that all lives in the world of front end development. Here's what you need to ask as you evaluate your code.

1. Are your pages and content marked up correctly? Is every tag used *semantically* and *meaningfully* (e.g. does everything that needs a tag have one, and is every tag used correctly when it's deployed?)
2. Does your pass a code test? [CodeSniffer](#) is a popular tool you can use to automatically review your code against different standards and will help you identify and fix accessibility errors.

3. Does your code sit in the order that the elements on the page? If someone reads your code top to bottom and draws your site based on that, is everything going to be in the right place?
4. Is your website responsive to different devices?

Using tools like [Google Lighthouse](#), [Axe](#) and [WAVE](#) can help you quickly and easily identify problems, and will even help you map them onto the WCAG frameworks that you're potentially violating. While this isn't the be-all to end-all for QA testing, it is a good place to start.

**TIP:** If you're looking for software suites and tools to help you with accessibility testing and remediation on an ongoing basis, there are a variety of options on the market, with pros and cons each. The [W3C maintains an ongoing list of resources and tools](#) to evaluate accessibility. Your team will likely have their favourites.

## Patterns and styles

First, patterns and styles. The WAI-ARIA requirements outline exactly what standard each pattern or style needs to meet, and those standards do map onto WCAG 2.0 requirements. The challenge, of course, is doing that mapping, and that exercise is where we see consultants or advisors offering a lot of assistance. Basically helping QA testers understand through all the noise of automated testing, what specific things need to be fixed.

If we go a level deeper, pattern and style QA is similar to front-end code testing. Angular, React and Vue include accessibility in their frameworks, getting your patterns off to a good start. Then, robust automated + manual testing will help take you through the next phase.

However, this is where there are a few things to watch out for.

First, make sure your pattern QA testing isn't happy path testing — you want to test for the worst-case user, not the straight-forward best-case one.

Second, make sure that you're applying a bit of common sense to the automated test design. We always recommend testers go and manually try the site without a mouse and using a screen reader. That will help inform better tests, and thus ship a better product.

And finally, make sure that you're testing different states of each pattern that has multiple states to test. You don't want one state to be fully accessible and the other states to be utterly useless!

## **Content types**

Finally, content types. So far we've talked about ARIA tags and the various rich-experience content types that you can make accessible using a framework. But there's something else to consider when you're QA-ing your site for accessibility: the content types of your more basic content.

HTML content types / media types include application, audio, for example, font, image, message, model, multipart, text and video. Then, each of these includes a sub-category (e.g. image/gif).

Content types help people with disabilities understand what they're looking at if they can't interact with your site.

So how do you QA this? Again, here are a few questions the QA teams should be asking themselves:

- Can users who rely on content type tags understand what content goes where?

- Can someone understand what goes where in your site just by looking at the HTML code?
- If you're using enriched media (videos, etc...) how are those going to be interpreted by someone with disabilities? Will they still get a usable experience?

The content types we choose, and how we represent them digitally, is absolutely essential to accessibility, but often does unconsidered as accessibility is considered a technical rather than a content puzzle to solve. Content producers need to work closely with development teams to make sure what they're creating is accessible, and also that what they're created is being deployed in an accessible way.

## User acceptance testing (UAT)

One of the final stages of the SDLC is end-user acceptance testing. Finally, the work for the team is put in the hands of actual users to see if they use it the way that it was built.

And testing for accessibility here is the #1 thing you can do to build a site that's both compliant and usable for everyone.

The goal is simple: test your site against a wide range of users *with a variety of abilities*. Just like you shouldn't just build test cases for the best scenario, you shouldn't build a user testing framework that only looks at one type of user.

If a fully-fledged user testing step isn't a part of your lifecycle development due to cost or time, then a good stand-in is to design a test case for yourself where you navigate through your site with a keyboard only and using a screen reader. This

obviously isn't perfect, but it gives your site a sort of common sense check to see if there are any glaringly obvious problems.

## Content testing

Content testing is the amalgamation of the automated testing you've completed to date. You want to test different pieces of content and the experience of consuming them across keyboard-only and screen reader-only sessions. The goal is to answer a few basic questions:

- Can you meaningfully consume the content of the site?
- Do you experience the content in the order that it's intended to be experienced in?
- Is it a good experience? Do you have agency over what content you're consuming without relying on a mouse or a thumb to scroll and pause?

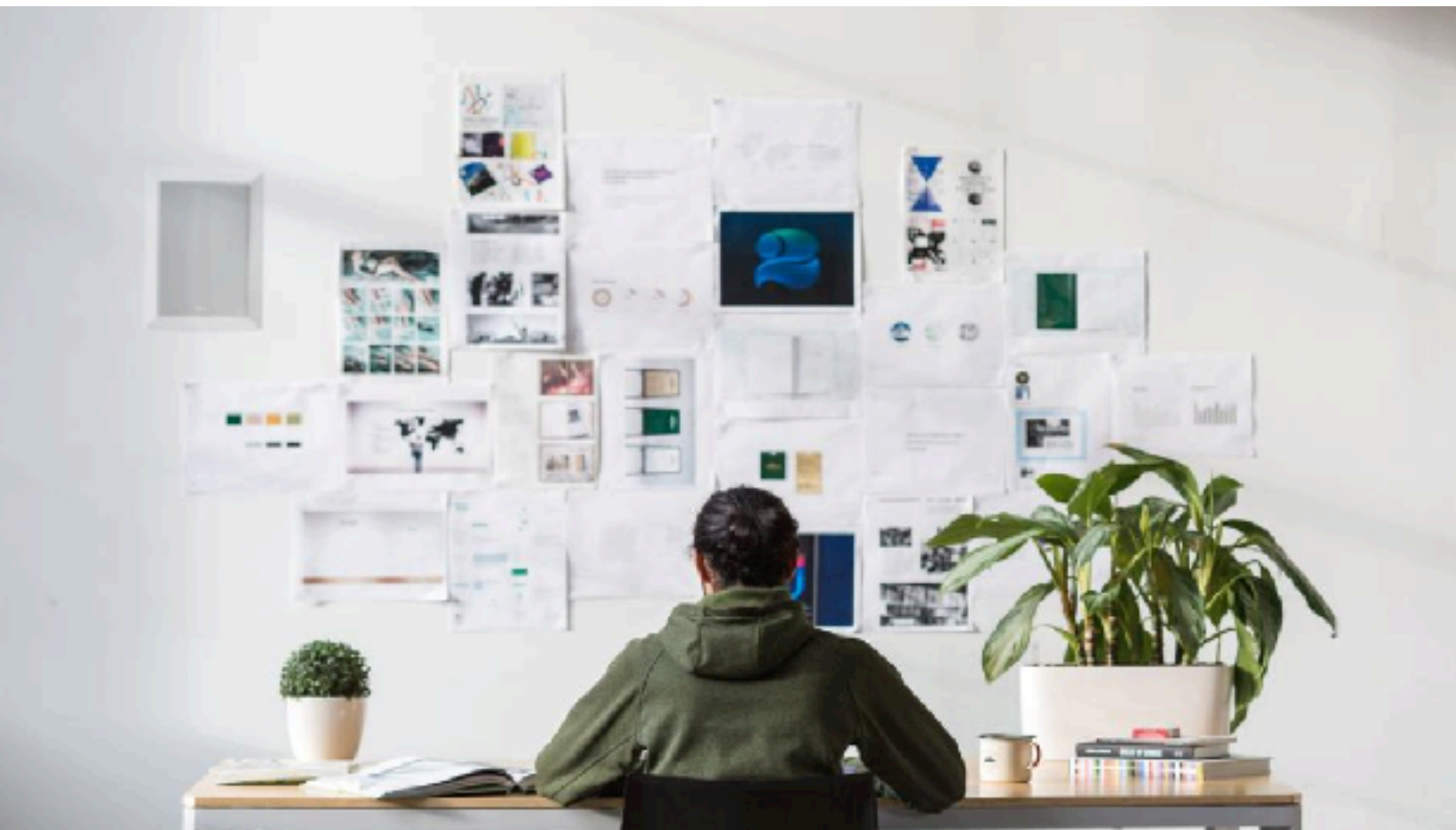
If these three things are all true, then your site is in a good position from a usability perspective, which in turn drives compliance with AODA and ADA regulations.

There are two common problems worth calling out here that are often missed.

First, alt tags. We've talked about alt tags before, but they're worth calling out again because of their importance. Alt tags are what tell screen readers what the content is if it's a piece of enriched content. For example, a picture of a dog would have the alt tag "picture of dog".

Alt tags are often abused for SEO reasons to get more keywords onto the page. However, this is a dated practice. While there might be some SEO value from the keyword, there's far more value from having a site that's responsive, secure, and accessible.





Second, video transcription. Video transcription does actually have a [specific WCAG 2.0 requirement](#) that maps into the AODA regulations, and as video becomes more prevalent, it's more important than ever to design and test with people with disabilities to make sure you're creating a positive experience. For pre-recorded videos, the expectation is to have transcripts to accompany your video. This is by far the most common use case for most organizations. Most videos won't need closed captioning, but, as so often happens with accessibility, there's increasingly a use case for closed captioning on social media. However, that is currently an optional rather than required piece for most videos. The same goes for audio.

## Chapter summary

- UX accessibility considerations include the patterns and interactions that you design as well as the technology you consider when you're building your website.
- A centralized pattern and interaction library help you document, replicate, and scale your accessibility over time.
- Graphic / visual design considerations include the basics like contrast, font size, and line height, but also patterns and interaction styles. One thing in particular to watch out for is where colour alone or visual alone is used to communicate meaning. Remember to have an additional indicator for both colour-blind users and those with visual impairments using screen readers.
- QA considerations are how you test for accessibility. Machine testing using tools like [CodeSniffer](#), [Axe](#), and [WAVE](#) can help. In addition, it's important to train developers to test their own code for accessibility, using the full range of HTML and WAI-ARIA attributes as needed.
- QA testing for accessibility also includes testing patterns and styles, and keeping both consistent, as well as testing various content types to ensure the meaning and functionality is the same across multiple experiences (e.g. keyboard only, screen readers, and mouse-and-keyboard sessions).
- User testing should be used throughout the SDLC with expected users and with those with disabilities. Experiencing or watching someone experience

your design with a screen reader is the best way to build empathy and understanding for how to design better.

## 04 Managing costs

So far, we've been talking about accessibility as something that teams should be doing not only because it will make your website compliant, but because it's a good thing to do from a UX, design, and business perspective.

But the reality is, there is a cost to this work. Designing an accessible product for the first time may take longer, require more testing and iterating, and usually, include external specialists. Redesigning an existing product can amount to nearly the same amount of work, or more. Even if you are a web design and development agency with ample internal resources, if a client requires an accessible website, unless you have the expertise on staff across both design and coding, it's often a good idea to partner with a consultant to help you navigate the project.



While we outlined the business case for accessibility in chapter 2, in this chapter we're going to look at the actual costs of managing an accessible project and a few options that product leaders have to achieve their goals.

By the end of this chapter, you'll be able to:

- Accurately project the incremental cost of accessibility into your project plan
- Identify and mitigate common project risk and sources of cost overruns
- Weigh the pros and cons of using internal or external resources
- Understand when to hire external help

## What does compliance cost?

There's no magic number that will get a site from non-compliant to compliant, and there's no magic number that you can tack on to your new build project for "accessibility". We can advise on baseline "starting at" costs however, which is better than nothing, which is often what is budgeted: zero.

The reality is, accessibility is an approach to design and development rather than a line item in a budget. And it does have a cost, particularly when you have to tackle it for the first time. It's like asking "how much will it cost to make my waterfall project an agile one" or "how much will it cost to use a Jamie Oliver recipe to roast a chicken vs my mom's family recipe".

The point is, there's no "one size fits all" answer. It will depend on:

- The existing content and code, and how much needs to be changed, added or rebuilt
- The presence and compliance levels of existing design patterns
- Any existing design or product language you might have in place

- How much content you have and how complex it is

With that in mind, we're going to bucket a few common scenarios to give you a sense of what you can expect to pay and what that money's going to buy.

**TIP:** Assume that an accessibility champion is on board from the very start. As we'll see, addressing accessibility halfway through a project - or after the project has launched - is more expensive and a lot more painful.

## Scenario 1: New build or redesign with few / none existing resources

This is one of the most common scenarios we see. Basically, a business or association needs to redesign or build new, realizing that:

1. Accessibility is important, and/or;
2. There is a requirement for AODA legally
3. They have no resources in the form of either staff with expertise, or reusable design patterns and code libraries.

This is probably where the cost is the most variable since where you start is going to have a big impact on how expensive it's going to be to get you where you need to go. In particular, your cost is going to be driven by:

- The number of pages and patterns you need redesign to be accessible
- The number and variety of interactive elements
- The types and volume of content that the site has in place
- The level of compliance the website owner wishes to achieve.



In our experience, smaller sites looking to achieve WCAG 2.0 AA compliance can expect an additional \$5K - \$10K +/- to achieve compliance. That's because the move from redesign to accessible redesign may be relatively simple.

Costs, however, climb quickly in line with the complexity of the site and the amount of content. For instance, say you have a content website that you're redesigning. You might need to add alt tags to every image, and your site might have thousands of images.

Even if you outsource this to a low-cost provider, it's still a hefty chunk of resources spent on pairing relevant text to images.

Likewise, if you have 100 layouts to redesign instead of 5 or 10, then the cost is going to increase significantly.

For larger sites, you're best to budget **at least \$10,000** as a discovery and recommendations budget to identify what compliance means to the project (and what's actually needed) and then a **10%-20% increase on top of your overall design, development, and QA testing costs throughout the SDLC.**



Oftentimes, we find ourselves helping SMBs in this position simplify their websites, patterns, and content to reduce the sheer volume of design and development work. Going back to our alt tags example, we might scrap half of the pages those images are on, streamlining the site and accessibility process. We might also scrap undesirable UX patterns in favour of simplified ones, such as single, static hero banner instead of rotating carousel.

## **Scenario 2: Upgrade existing site to meet accessibility standards**

What also happens is accessibility emerges as a strategic initiative for the organization, and accessifying the website becomes part of that mandate.

This approach has pros and cons.

On the plus side, a lot of work is usually already done. Especially if the site isn't too old and is built on top of an accessible theme or framework, then it may be possible to make relatively quick changes to the code and design that make the site far more accessible.

However, what can look like a small tweak from the outside can quickly spill into an incredibly expensive project.

If the site's relatively large and has been custom coded, then refactoring, redesigning, and testing will likely be a significant cost.

For instance, say you're running an ecommerce site. It's pretty straight forward, not that many page templates and flows, and only a few complex interactions.

But let's dive a little deeper. To be made accessible, you'd have to:

- Fix all your content (alt tags, transcripts, etc... )
- Review all your third party integrations



- Do a complete audit of your site to identify specific accessibility violations, then dig through the code to find the cause and update it at its source
- Design accessible, navigable pages and patterns for every interaction you have on your site, including adding to basket, checking out, and completing your order form / inputting information.

Even if the design or pages will remain much the same, every piece of code needs to be audited, tested, and re-factored. Sometimes this means adjusting core modules for content outputted by the systems that generate your front-end pages (i.e. your CMS, CRM, e-commerce platform) OR a procurement exercise to identify a new technology solution, followed up by a migration project to rebuild from scratch.

Given the above, it's unlikely that internal resources alone can execute on all the recommendations, or even know where to start with the recommendations from an expert. And sites don't even have to be that complex to end up needing substantial work to achieve WCAG 2.0 AA compliance. In those scenarios, accessibility can end up costing significantly depending on how complex and how vast your product, app or website is.



**TIP.** Want to learn more about choosing the right technology? [Visit our Resources](#) to find related ebooks on technology procurement.

## Where compliance dollars are spent

Given the above scenarios, it's unlikely that making your website accessible is going to be something that is taken lightly. So with that in mind, where do compliance dollars end up being spent?

There are four major buckets to consider: content fixes, code refactoring, and pattern / element redesign, and testing.

### Content fixes

Content fixes include things like adding alt tags, transcribing videos, adding closed captioning, and fixing hyperlinks so they make sense. Content fixes can also include helping change what the architecture of the site is labelled, if not the underlying architecture itself (e.g. 'get in touch' instead of 'contact'). There may be more specific content work that needs to be done, like converting PDFs into accessible web pages.

Content fixes are generally on the simpler end of the spectrum, and the main cost factor is volume: it's more expensive to transcribe 10,000 videos or mark up 10,000 accessible PDFs, than 1,000 or 100. The best strategy here when dealing with large volumes is to convert by priority (most - least used, or newest to oldest, or as needed).

## Code refactoring

Code refactoring is probably the largest and the most variable cost when it comes to an accessibility project.

Since almost no developer writes code completely from scratch, but rather repurposes code from libraries, forums, and their own previous experiences, the amount of refactoring needed to meet accessibility recommendations can vary wildly. Whether it's a site that's been built relatively recently with relatively few hands involved in the code or an older site that has had multiple owners bolting on various functions over the years, if there are no design guidelines or centralized patterns, then the cost of refactoring can be high, as simple accessibility changes require a lot of effort.

## Pattern / element (re)design

Pattern and element design are capital-intensive projects. It takes time to build up a library, and that time costs money. If you're planning on continuing to develop additional products and web properties over time, then investing in this is a good idea. However, if your website is truly a one-off investment, then the patterns and templates you build won't have the same long-term value. If you have some internal development resources, a third party can be extremely helpful in designing the right pattern for you to match, or help you select a pre-built theme that already does most of the work. Basically, it's a lot easier to achieve compliance if you have a firm blueprint for how you need to get there, and are confident that it's correct.

The cost of pattern and element design (design systems) are a combination of volume (e.g. how many patterns are you designing) and complexity (how hard is the problem to solve). The more patterns you need, and the more complex they are, then the more it's going to cost.

## Testing

Finally, testing. Testing a site for accessibility is, as we mentioned in the last chapter, a combination of both machine / manual testing, QA testing, and usability testing. The main cost here is time and resources, as well as the subsequent cost of work that needs to be redone. The easiest way to control testing costs is to do it early and often in the SDLC:

- Train UX and graphic designers to design with accessibility from the get-go
- Train developers to test their own code
- Train QA testers on accessibility, so they catch more things before you go to usability testing
- Work in shorter iterations rather than one big push and test as you go

## Common cost spirals

### Getting accessibility involved after design

The biggest cause we see for expensive accessibility projects is when accessibility is considered an afterthought late in the design process. It's realized that compliance is required, and organizations are forced to change or completely redesign their graphic and user experience design, and then change the code accordingly. If the content services platform delivering the front-end experience is generating structural non-compliant code, your team(s) may have to consider a procurement and migration exercise as well. This can add costs upwards of 100% of the original budget for the planning and design phase(s).

### Not spending early to train developers and QA testers on how to do accessibility

QA testing is often seen as an objective role — something either passes a test or it doesn't. Accessibility requires a lot of shades of grey. It takes someone understanding what the desired outcome is, understanding the frameworks that

guide decisions and then making decisions accordingly. When QA professionals don't have accessibility training, this becomes much more difficult, and invariably accessibility QA ends up being outsourced to consultants at a premium hourly rate. Rates vary for local vs. offshore, but you can budget for \$65/hour - \$120+/hour CAD depending on the seniority level.

## Failing to prioritize

If you dig through the WCAG 2.0 guidelines, you'll quickly be overwhelmed by the myriad of details. Organizations who try and fold accessibility into their own teams can fail because they don't know which bits are critical, important, or irrelevant, and compensate by trying to do everything - or nothing - well. This can slow development to a crawl, increasing the cost substantially for an experience that is only marginally better.



## Internal vs external resources

Most teams don't have a dedicated, internal accessibility resource for the simple reason that most organizations haven't prioritized it. Accessibility is an ongoing project, but one that can (and should) be embedded in ways of working for each team. Designers should design with accessibility in mind, developers should develop with accessibility in mind, and QA teams should know how and what to test.

All the "shoulds" aside, that does mean that when it comes to executing on a significant change like getting a website up to WCAG 2.0 Level AA standards, you simply may not have the domain expertise available internally to do the job.

There are three kinds of external resources you can hire: people who aren't trained in accessibility, accessibility domain experts, or accessibility consultants. We're going to look at the pros and cons of each.

### Untrained help

You might hire a freelance developer, content writer, or designer to help you as you move through the project.

#### Pros

- If you already know what you need and just don't have the bandwidth, then this is a great option, to increase capacity.

#### Cons

- May be unable to help with specific accessibility questions
- May design in a non-compliant way

- Requires extremely detailed instructions to meet your standards.

This is good for when you know what you need to do and are simply looking for the bodies to get it done. It could cost you extra time in oversight and QA...

## **Accessible domain expert**

This is a resource (individual or services firm) who specializes in accessibility generally, or specifically (e.g. PDF content remediation, QA testing, front-end development...).

### Pros

- Great if your team has a good understanding of principles and everyone has some level of training, but you're finding yourself lacking in 1-2 key areas.
- Can help you solve tricky problems if you're generally compliant but have a few areas where you need help (e.g. understand compliance frameworks, but need some help QA-ing to meet WCAG standards).

### Cons

- Requires you to have a good big-picture view of compliance requirements
- Isn't able to unite teams across a wide range of disciplines in a way that's required for compliance delivery.
- May be a project management / resource management challenge if you don't have the budget to involve them tightly through the SDLC

This is good for organizations who have a fairly good grasp of accessibility requirements, are struggling with a specific area, have a specific technical problem they're hoping to solve, and/or know when and how long to involve specialized resources at specific points in the timeline.





## Accessibility advisor (ongoing)

This is essentially an end-to-end accessibility service. Typically a firm but could be an individual, they have significant expertise in accessibility and can help you decide what to do and the best way to do it. Regardless of your project size, this should be a key advisor to consider, because even a small partnership can get you headed in the right direction.

### Pros

- **Expertise in accessibility.** Can help you decide what to do and how to do it
- **Training.** Can help make your team more accessibility -minded, resulting in self-sustaining compliance for the long term
- **Troubleshooting.** Can help you tackle thorny pattern and interaction problems as effectively as possible.
- **Prioritizing.** Can help prioritize WCAG requirements to focus your energy where it needs to go to get maximum accessibility bang for your buck.
- **Scalability.** Can dedicate significant, trained resources to you against your project to keep it on track

### Cons



- **It's difficult to pick the right one.** We recommend choosing someone with both experiences in your industry and experience with your website size / scope.
- **It can breed dependence rather than learning.** You might have to keep going back and never learn to build or maintain anything on your own.
- **Cost.** External help can be expensive, and getting buy-in may be challenging, especially if they're making recommendations rather than coding solutions.

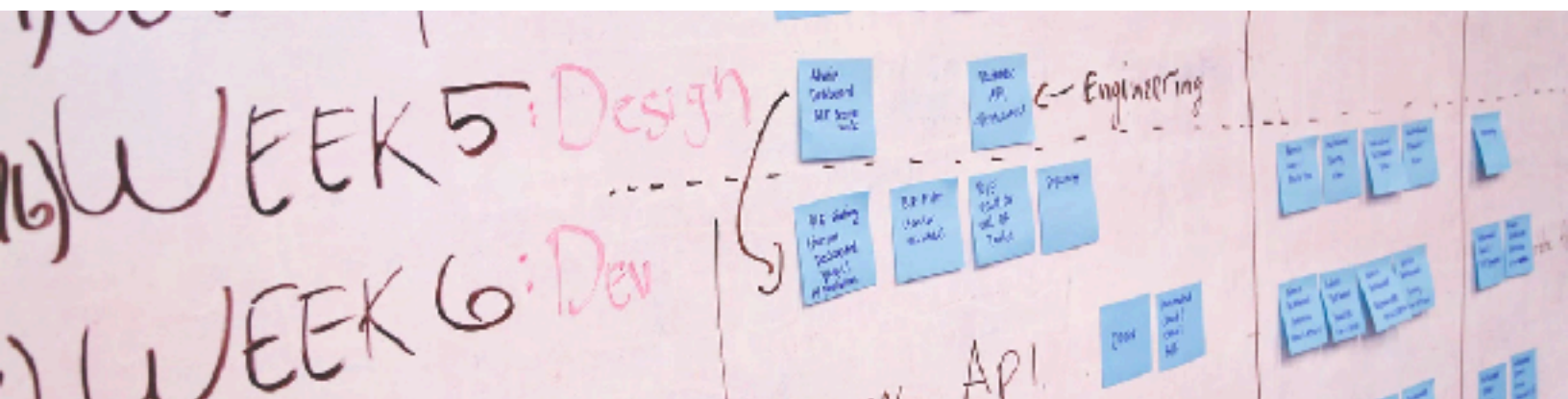
## Chapter summary

- To achieve accessibility compliance, most sites will cost a minimum of \$5,000 - \$10,000 upfront. Larger sites or sites with more complex problems will need to add 10% - 20% to the total SDLC cost until/unless the design and development process is completely inculcated within your production team. Smaller sites, or those with almost no issues and relatively low refactoring costs, can likely get away with extending time frames rather than dramatically increasing spend.
- Accessibility spend tends to concentrate in 4 main buckets: pattern and component design (UX/UI), content accessibility, code refactoring, and testing. Content tends to be solely volume, while the other three's pricing is reflective of both volume and complexity.
- Common sources of cost overruns are getting accessibility involved too late, failing to properly train the QA team, working with design and/or dev resources who are not well trained, failing to budget properly, failing to prioritize WCAG 2.0 standards, and outsourcing tactically rather than strategically.
- Knowing when to outsource is essential to keep your project moving forward. Not only is it critical that you understand what different vendors can do for you, but you need to understand what you're looking for — do you want a partner, or just need someone to put in the hours? Outsourcing to meet the needs of the project is essential to project success.

You're done, right?

**Once and Done:** Push everything you have into your launch, and then essentially

In this chapter, we're going to explore these two options, their pros and cons, and a few common use cases for each.



## Launch “once and done”

The first approach is heavily focused on the launch of a website. The idea is that all the testing, evaluation, content updating, and remediation is all completed *prior to the launch*. When you turn everything on, you're done. Ongoing maintenance with regard to accessibility is limited if it happens at all.

### Pros

- In general, costs are easier to account for and return on investment is easier / more simplistic to track.
- Your website is wholly compliant to the standards you set (at least for the time being)
- You can be assured you've reached a level of compliance by a certain date — essential if your level of legal requirement changes over time
- Your executive sponsors and legal department can tick the box that the requirement was delivered

### Cons

- This approach requires much more cash upfront, since everything has to be completed
- Certain elements tend to get dropped as the project continues
- It exposes you to additional project risk (for example, designing something that your users don't like or finding serious problems in your code but not having the time or refactor it correctly)
- As site content, features and functions are created, updated, and evolve, eventually you'll be non-compliant again

## Use case

The “once and done” approach is useful in a couple of instances. For those with very limited internal resources, it can be helpful in containing costs and project scope. Plus, if you’re not in a regulated industry, the requirements are fairly fixed, so if the current-state is correctly understood at the start, then this can be a great approach.

This approach also works well for project owners who are heavily reliant on external resources like freelancers and consultants. If you can scope exactly what’s needed and when, you can often negotiate a better rate, helping to bring costs in line with your budget.

The biggest issue is your compliance will degrade over time. This re-introduces the risks we talked about in [Chapter 2: The business case for compliance](#). If you let your site go for too long, in only a few years you’ll have to start over again.

## Ongoing retainer / program

The second approach to launching is to do it incrementally. Basically, you take your project and prioritize based on importance and completion speed. That becomes your backlog, and you tackle your progress in multiple steps or releases.

Normally, this will culminate in some sort of soft launch when a major change or the majority of changes go through, but there’s ongoing work indefinitely on your site. Ongoing work might be continuing to work through a content backlog, or continually refactoring your code. This approach generally aligns with a more agile design and development approach.

## Pros

- Allows for ongoing maintenance and improvement, usually resulting in a better project over time
- Makes it easy to stay up to date with changing compliance standards
- Content is always being created, so there's a constant need for accessibility checks
- Creates an environment that allows for ongoing testing and evaluation
- Leads to better knowledge transfer and education to specialists and administrators

## Cons

- Requires continuous investment, upwards of \$1,000 - \$3,000 a month plus ongoing time commitments for testing resources (staff and/or software to do this)
- Requires internal or external accessibility resources

## Use case

In general, we recommend the ongoing maintenance approach for most organizations. In particular, heavily regulated industries, government, or associations should consider this approach because it's much more flexible in response to changing compliance regulations that require ongoing testing, evaluation, and adjustment. At the same time, you continually improve your capabilities to build and design in a compliant way since it's an ongoing project.

For organizations that want to bake accessibility into their culture and their way of working, an ongoing commitment is the best way forward because it means that accessibility isn't some "other" project. It's just how all projects are done.

## Chapter summary

- There are two main approaches for post-launch maintenance: the “*once and done*” approach, where all accessibility work is completed within the scope and timeline of the project, and the *ongoing maintenance* approach, where resources are invested continuously over the long term for continuous improvement.
- The “*once and done*” approach helps organizations control for cost and time, presuming the project is accurately scoped and there are few unexpected challenges.
- For relatively simple projects, the “*once and done*” approach works extremely well. However, it does leave organizations vulnerable to regulatory changes, project risks like major roadblocks, and tends to transfer knowledge from external experts (if consulted) to internal resources.
- *Ongoing program* is usually used by organizations who either are making accessibility a core priority / value, or those in heavily regulated industries for whom compliance is mission-critical.

*Ongoing program* will generally start around \$1,000 - \$3,000 per month, along with significant time from internal resources. However, you do get a better result in the long run, content and user experience that's continuously more compliant, and you can foster an environment for ongoing testing and evaluation.

## 06 Common risks and pitfalls

We do a lot of accessibility projects. And over the years, we've seen plenty go smoothly. But we've also seen everything that can go wrong. So in this final chapter, we put together a list of the most common risks, pitfalls, and project roadblocks we've seen derail website compliance — and how to avoid them.

### **CMS isn't compatible with accessible patterns and design**

Most CMS software today can and should support accessible front-end design and code patterns.

Most, but not all.

A less frequently encountered but showstopper cause for accessibility projects going off the rails is when a CMS isn't compatible and has to be completely ripped out and replaced, or it's core re-developed / upgraded.

A CMS migration - or core tune-up - is an entirely new project on top of the accessibility and compliance one you may have scoped. If your CMS is not going to support accessible patterns, you'll have to address this first and foremost.

#### **What to do**

The best solution that we've seen is to simply separate the projects into their components — the accessibility and compliance project, and the CMS upgrade or



migration. This is yet another great example of why it's so important to involve accessibility early in the SDLC as well. It's entirely possible for a CMS to work perfectly for the rest of the organization while being fundamentally broken for accessibility specialists.

By splitting the projects — and even shelving the accessibility project until the CMS is dealt with, makes it far easier for product owners.

**TIP:** Want to learn more about choosing the right technology? [Visit our Resources](#) to find related ebooks on technology procurement.



## Alienating your existing users

A redesign of a website is always a risky proposition in the sense that any major change is likely to alienate at least some of your users.

Just think about how outraged users get when their favourite social media platform interface gets an update.

The same can happen when you're designing for accessibility. This is usually a problem when organizations take an existing design pattern and attempt to shoehorn in accessibility features when they're not appropriate. The result is malicious compliance — yes, it technically meets AODA / ADA standards, but it fails to address the overarching goal of *making your website easier to use for everyone*.

### What to do

- **Consider function over form.** Make sure that your accessibility design doesn't sacrifice the function of your site for users just to preserve the form. People will get over a site that doesn't "look great"...visual design is subjective. They won't get over a site that doesn't work.
- **Talk to your users.** Usability test your existing users and see what they think. Does the proposed design still meet their needs? Does it still allow them to use the site as they want, not as you design?
- **Offer embedded help tools and support guides.** Chatbots, intelligent support forums, and other types of online support tactics can help guide your users through your site and teach them where to find or how to use content and features.

- **Deploy product analytics and insights tools.** There is a lot of software on the market that will track user behaviours, analyze conversion goals you set, and help you understand what users are doing (or aren't doing) on your site. These tools, with the right analysis, can provide deep insights into your ongoing improvement effort, and help set priorities in your product / CX / UX roadmap.

**TIP:** Set up for relevant, ongoing success by providing qualitative and quantitative feedback measurement. Don't guess; gather data from users.

## Legal ramifications of making a mistake / inaction

We talked about project risk earlier in Chapter 2, but this is an increasingly common failure for organizations considering accessibility.

At the time of writing, no web-based AODA violations have been pursued by the Ontario government.

However, there have been over [2,000 lawsuits filed in 2019 alone for ADA violations](#).

So the risk of failing to act is not abstract, but increasingly, can be effectively measured in dollars and cents.

What's more, most of these cases are not about a website update to AODA / ADA compliance that failed to achieve its goal. Rather, they're businesses who didn't act, and now are facing the consequences of those decisions.



Generally speaking, these lawsuits result in a legal order to fix the offending web property. But when you add up the legal fees, brand equity spent, and the potential cost of complying with an investigation, it's an awful lot cheaper to update your website ahead of time. At the same time, you can face significant fines or have to pay a settlement — both of which will cost you.

## What to do

- **Get ahead of the curve.** Start updating your website to meet accessibility requirements sooner rather than later. Demonstrate your commitment to AODA and be honest and transparent about your conformance level.
- **Identify common threads.** Look at AODA / ADA lawsuits and pull out common threads (e.g. incompatibility with screen readers) and prioritize those fixes for your own site.
- **Be honest about your compliance and conformance level.** Include an accessibility statement on your site that is transparent and the truth. [W3 offers a fantastic tool for developing an accessibility statement](#), use it and have your legal team review for any “gotchas”.

## Failing to design a common design language

During a website build, it's often tempting to switch coding and designing *right* for coding and designing *fast*. What happens is, to meet an aggressive deadline, organizations focus on executing — get wireframes, mock-ups, and code out the door as quickly as possible, rather than spending the time to build and document clean, repeatable assets, and patterns, or define a common design language for how problems are tackled, designed, and deployed.

The result is a website that from the outside looks and functions fine, but on the inside is extremely difficult to actually manipulate, change, or update. This causes all sorts of headaches as organizations either incrementally roll out accessibility, each time having to do ad hoc updates rather than systematic pushes, or leads to stagnant sites that get a major facelift periodically rather than continuously improving.

### What to do

- **Define an accessible design language early.** Partner or hire an accessibility expert to help you build and outline your design language. This should consist of modular patterns and interactions that can be reconfigured and reused again and again while still meeting AODA / ADA standards.
- **Define accessible coding practices.** Build processes and standards to turn your design library into reality without ad hoc customization. Build a centralized, modular repository for your code where the same tags, ARIA attributes, and classes govern the same elements across your website.
- **Educate internal teams on achieving accessibility.** Make sure all teams understand what accessibility is, what it means for their role, and how to do their jobs in a way that fosters accessible design.

## Failing to account for the total cost of ownership of one-off costs

The total cost of ownership (TCO) is a cost estimate that is often overlooked and undervalued. As organizations, we tend to be very bad at documenting and then adding up the total cost of doing a project a specific way, and even worse at accounting for the opportunity cost of a specific approach.

However, that's exactly what needs to happen when it comes to accessibility.

Here's a comparison of two accessibility scenarios.

First, an organization that doesn't consider total cost of ownership. They might not do an initial consultation with an expert and instead jump right into design and development. Say they invest \$10,000 for a bunch of ad hoc fixes. Then, a new person starts, who identifies a bunch more fixes for another \$10,000. Then, the organization realizes that it missed some requirements between the two updates, so opts for an additional \$10,000 to plug the gaps. Finally, realizing that there are



still some changes to make, the organizations has to hire a whole accessibility team for a complete audit and review (\$20,000) and then has to rip out and replace most of the ad hoc patches they'd done over the years for a centralized, organized, and shared accessible design language (\$15,000). **Total accessibility project cost: \$65,000.**

Here's scenario two: an accessibility consultant is hired to help build a structured roadmap to an accessible site (\$20,000). They stay on to advise (\$3,000 per month for 12 months = \$36,000) on the design, construction, and deployment of a fully accessible site, as well as train internal teams on accessibility. **The total cost of an accessible site: \$56,000.**

The point is that while scenario 1 had lower prices at each stage, it ended up costing 14% more overall because the total cost of ownership wasn't forecast and considered from the start.

## What to do

- **Plan and invest proactively rather than retroactively.** Make accessibility a priority from the start, rather than responding to changing regulations or events.
- **Calculate the total project cost upfront.** If you're not sure, then an accessibility advisor can be a good investment to identify where you are and what needs to be done from an accessibility perspective.

## Investing in low-impact compliance issues

One of the most common issues that we see when organizations tackle accessibility themselves is investing heavily on low-impact but easy-to-understand and easy-to-execute accessibility and compliance issues. For example, an

organization might spend weeks updating hyperlink text to be more intuitive, when their site is fundamentally unreadable by screen readers and isn't navigable without a mouse.

## What to do

**Conduct an accessibility audit and prioritize based on the impact and effort.** Start high-effort / high-impact projects early, identify high-impact / low-effort wins you can score along the way, and deprioritize everything else.



Image Credit: [Impact Effort Matrix](#).

## Trying to force every change into a single project

Finally, the biggest failing we see is when organizations try to force every accessibility change into a single project in “once and done” style. The reality is, accessibility is and will be a continuing and ongoing challenge. There’s almost no way to hit every accessibility milestone in one go, and when organizations try, project cost and timelines may spiral out of control.



## What to do

- **Treat accessibility as a roadmap.** Build an internal understanding of where you need to go, then prioritize your work into a series of sprints based on your impact matrix.
- **Start with a single major push.** It's often good to get a chunk of accessibility work done early to secure buy-in for additional investment. Double the length of your first sprint and tackle major accessibility challenges. Then, focus on ongoing investment in accessibility over time.
- **Focus on ongoing investment.** Build a culture, team capacity, common language, and common coding practices around accessibility so that you can support ongoing compliance changes.



## Chapter summary

- If your CMS isn't compatible with accessibility requirements, you may be better off replacing it rather than spending huge cash on intensive customization.
- If you do need to replace or upgrade your CMS, it's best to treat that as a separate procurement and migration project to your accessibility compliance project.
- There is a risk of alienating your existing users as you push through changes. Make sure you consult and test with existing users during the process, and educate users with on-site guides after the fact to help keep engagement high.
- Become familiar with the legal requirements you're obligated to meet, and prioritize hitting those major milestones. Getting a head start on your commitments and establishing a process with ongoing effort can mitigate and soften legal risk.
- Create a common design language and a centralized code system that is accessibility-compliant. This allows for modular page and layout construction without worrying about accessibility issues, and will make maintenance, changes and updates much easier than dealing with custom, one-off code. If you are running a multi-site deployment, think about how such issues replicate across multiple domains!

- When you're pricing your project, consider the total cost of ownership. Acting reactively, while you end up with each bill being lower, inevitably you'll end up spending a lot more money.
- Prioritize high impact compliance issues, even if they're difficult to deploy, over easy to understand quick fixes that don't fundamentally shift how easy your site is to use for disabled people.
- Build a schedule of ongoing changes rather than forcing everything into a single push. Accessibility is an approach to design and development, rather than a point project that can be crossed off a to-do list.

## 07 Final thoughts

We've come to the end of our accessibility journey together. Hopefully, by now you have a good idea of how to execute your own accessibility project, and how to deliver the user experience your website visitors expect, regardless of their situation.

And while we're almost done here, we hope that your journey is just starting. Karl Groves, the prolific accessibility advocate, [wrote in 2011 wrote](#):

*"When [accessibility] becomes part of how you do things, of course it is free."*



First, we built websites. Then Google prioritized websites that loaded fast. So we built faster sites. Then we started building responsive sites, and now mobile-first. Today, you would never ask "how much is it going to cost to make this work on my iPhone 11?" And that's where we see accessibility going.

Accessibility is the next evolution of web design. Eventually, it's not going to be a specific project, nor are there going to be accessibility experts any more than there are mobile-first design and development experts.



When something becomes baked into the process and way you work, the cost of that something goes down. What's more, as something moves from a "nice to have" to a "must have", infrastructure, training, services, and technology all spring up to help deliver on the new requirement.

Furthermore, with constantly rising user expectations around performance and ease of use, the requirements for accessibility are moving closer to the minimum viable requirements for any successful website deployment.

All this is incredibly exciting, and quite some time in the making. In 1999 the Web Accessibility Initiative, a project by the World Wide Web Consortium (W3C), published the Web Content Accessibility Guidelines WCAG 1.0. In 2008, the WAI released the WCAG 2.0 as a Recommendation. In 2018, the WCAG 2.1

extension was released. But until only a few years ago, accessibility was something that only bothered governments. Today, it's a priority for any website owner.

We're excited for the day when we don't have to build a business case for accessibility any more than we have to build a business case for having a website at all.

In the meantime, we're here to help. Good luck on your accessibility journey,

Simone Abel,

Director of Digital Strategy  
Enginess.io



## Need help with accessibility or AODA?

If you have any questions about your accessibility, ADA or AODA, get in touch with us using the contact information below.

We'd love to hear from you.

[info@enginess.io](mailto:info@enginess.io)

416.901.6151

[enginess.io](https://enginess.io)