

TERM PAPER INPUT

Ajax: A New Approach to Web Applications

Article written by Jesse James Garrett on February 18, 2005

If anything about current interaction design can be called “glamorous,” it’s creating Web applications. After all, when was the last time you heard someone rave about the interaction design of a product that wasn’t on the Web? (Okay, besides the *iPod* or *iPhone*.) All the cool, innovative new projects are online.

Despite this, Web interaction designers can’t help but feel a little envious of our colleagues who create desktop software. Desktop applications have a richness and responsiveness that has seemed out of reach on the Web. The same simplicity that enabled the Web’s rapid proliferation also creates a gap between the experiences we can provide and the experiences users can get from a desktop application.

That gap is closing. Take a look at *Google Suggest*. Watch the way the suggested terms update as you type, almost instantly. Now look at *Google Maps*. Zoom in. Use your cursor to grab the map and scroll around a bit. Again, everything happens almost instantly, with no waiting for pages to reload.

Google Suggest and Google Maps are two examples of a new approach to web applications that we at Adaptive Path have been calling Ajax. The name is shorthand for **A**synchronous **J**avaScript and **X**ML, and it represents a fundamental shift in what’s possible on the Web.

Defining Ajax

Ajax isn’t a technology. It’s really several technologies, each flourishing in its own right, coming together in powerful new ways. Ajax incorporates:

- standards-based presentation using XHTML and CSS;
- dynamic display and interaction using the Document Object Model;
- data interchange and manipulation using XML and XSLT;
- asynchronous data retrieval using XMLHttpRequest;
- and JavaScript binding everything together.

The classic web application model works like this: Most user actions in the interface trigger an HTTP request back to a web server. The server does some processing — retrieving data, crunching numbers, talking to various legacy systems — and then returns an HTML page to the client. It’s a model adapted from the Web’s original use as a hypertext medium, but as fans of The Elements of User Experience know, what makes the Web good for hypertext doesn’t necessarily make it good for software applications.

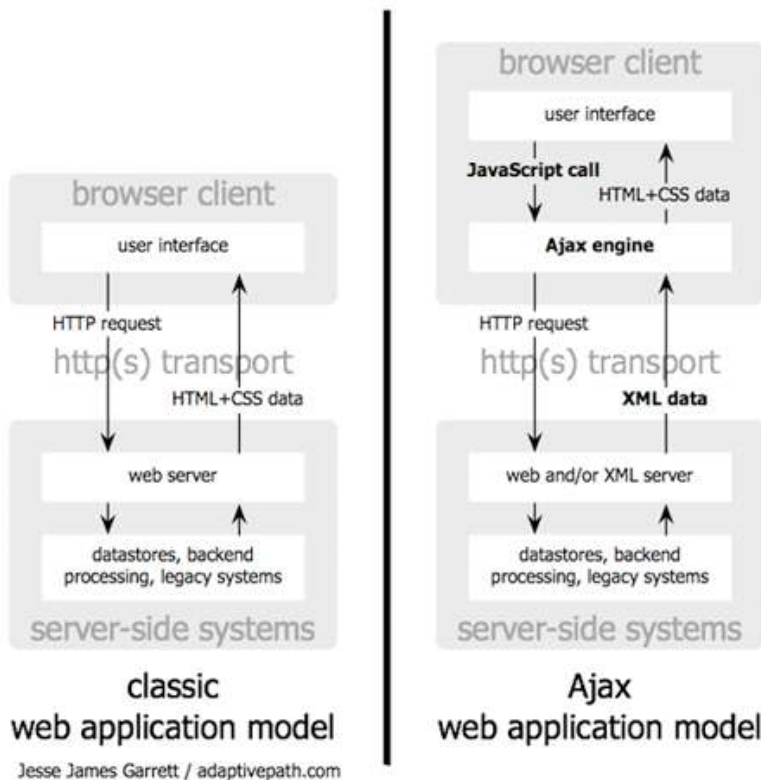


Figure 1: The traditional model for web applications (left) compared to the Ajax model (right).

This approach makes a lot of technical sense, but it doesn't make for a great user experience. While the server is doing its thing, what's the user doing? That's right, waiting. And at every step in a task, the user waits some more.

Obviously, if we were designing the Web from scratch for applications, we wouldn't make users wait around. Once an interface is loaded, why should the user interaction come to a halt every time the application needs something from the server? In fact, why should the user see the application go to the server at all?

How Ajax is Different

An Ajax application eliminates the start-stop-start-stop nature of interaction on the Web by introducing an intermediary — an Ajax engine — between the user and the server. It seems like adding a layer to the application would make it less responsive, but the opposite is true.

Instead of loading a webpage, at the start of the session, the browser loads an Ajax engine — written in JavaScript and usually tucked away in a hidden frame. This engine is responsible for both rendering the interface the user sees and communicating with the server on the user's behalf. The Ajax engine allows the user's interaction with the application to happen asynchronously — independent of communication with the server. So the user is never staring at a blank browser window and an hourglass icon, waiting around for the server to do something.

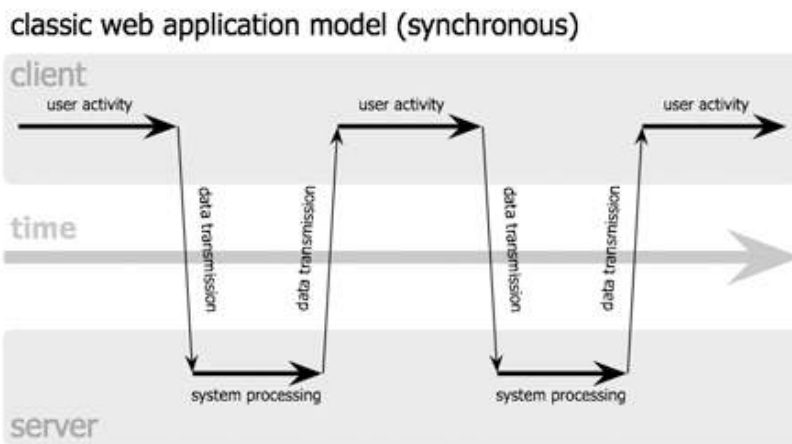
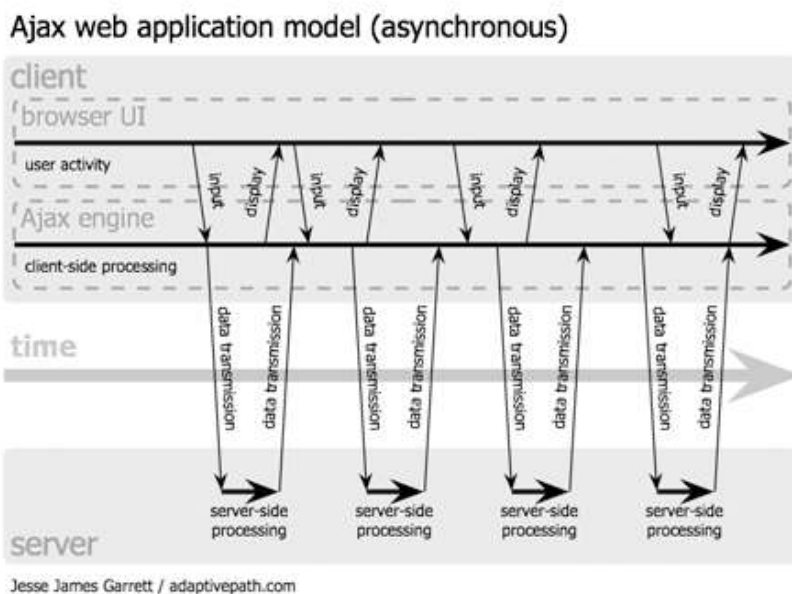


Figure 2: The synchronous interaction pattern of a traditional web application (top) compared with the asynchronous pattern of an Ajax application (bottom).



Every user action that normally would generate an HTTP request takes the form of a JavaScript call to the Ajax engine instead. Any response to a user action that doesn't require a trip back to the server — such as simple data validation, editing data in memory, and even some navigation — the engine handles on its own. If the engine needs something from the server in order to respond — if it's submitting data for processing, loading additional interface code, or retrieving new data — the engine makes those requests asynchronously, usually using XML, without stalling a user's interaction with the application.

Who's Using Ajax

Google is making a huge investment in developing the Ajax approach. All of the major products Google has introduced over the last year — Orkut, Gmail, the latest beta version of Google Groups, Google Suggest, and Google Maps — are Ajax applications. (For more on the technical nuts and bolts of these Ajax implementations, check out these excellent analyses of Gmail, Google Suggest, and Google Maps.) Others are following suit: many of the features that people love in Flickr depend on Ajax, and Amazon's A9.com search engine applies similar techniques.

These projects demonstrate that Ajax is not only technically sound, but also practical for real-world applications. This isn't another technology that only works in a laboratory. And Ajax applications can

be any size, from the very simple, single-function Google Suggest to the very complex and sophisticated Google Maps.

At Adaptive Path, we've been doing our own work with Ajax over the last several months, and we're realizing we've only scratched the surface of the rich interaction and responsiveness that Ajax applications can provide. Ajax is an important development for Web applications, and its importance is only going to grow. And because there are so many developers out there who already know how to use these technologies, we expect to see many more organizations following Google's lead in reaping the competitive advantage Ajax provides.

Moving Forward

The biggest challenges in creating Ajax applications are not technical. The core Ajax technologies are mature, stable, and well understood. Instead, the challenges are for the designers of these applications: to forget what we think we know about the limitations of the Web, and begin to imagine a wider, richer range of possibilities.

Separation Anxiety: The Myth of the Separation of Style from Content

Article written by Bob Stein on November 17, 2000

I've read several HTML references, online and off, and all seem to make some mention of the dichotomy of style and content, presentation and structure, appearance and substance. The good designer is admonished to keep them separate in order to ward off various woes: unmanageability, unusability, professional shame.

I think this is a myth. I think it has persisted for four different reasons. I don't think web designers need be concerned about it at all.

Vital to good web design is architecting data so the site is usable and maintainable. This begins only after a thorough grasp of mission and means. A textbook web site may fit together nicely with one simple separation, style from content. A real site begs deeper insight and benefits from separations based on richer reasoning.

Separating style from content is like the training wedge in skiing, practiced in the first lessons, abandoned with a little skill. Maybe it's helpful in the introduction. Soothe initial confusion with a simple anchoring idea. Give fretful consciousness something to worry on while the intuitive subconscious practices the real work. I suspect the student is best served by moving promptly beyond. Good design strategies would be internalized better by analyzing living web sites, discussing real how's and why's, understanding the shameful compromises, and speculating on practical improvements.

What I think working web designers need is visibility into the technology they're trying to use and into the experience they're creating for their users. Combine that with a thorough grasp of mission, and an inclination to complete the job, and they'll decide wisely what needs to go where. Down in the trenches, it's vastly more complicated than separating style and content.

The myth of separate style and content as a design strategy persists because a lot of great web sites look as if they were designed that way. It's very hard to isolate what makes a site thrive. That one aspect is common and observable and so gains mindshare.

A well-organized useful site may very likely separate a lot of content from its presentation. A database of well-structured content here, a series of stylish and useful presentation media there. But to call style/content separation a major aspect of web design strategy is like calling mud a major strategy of war.

Any observer can see that war makes a lot of mud, but that doesn't really describe what war is doing. Nor is it something soldiers need talk and think a lot about. Good soldiering means aiming mortar to kill the other team, and digging trench to protect one's own. This may make a great deal of mud, but those in the trenches on either end of a missile trajectory are focused elsewhere.

A thriving web site may have a lot of separate style and content. But that doesn't describe at all what the designers have done. Good design means organizing data and ongoing maintenance of the site so they are handled efficiently by employees and the results are useful to users. This is a vastly complex challenge, and if you take it on thinking the separation of style from content holds great power, someone is in for disillusionment (you if you learn a lot, your client if you don't). At many points of a project, designers may isolate presentation from structure. But their focus is elsewhere, on specific concerns that aren't useful to discuss too broadly. Anyone outside the problem is never going to get it, never going to judge accurately.

There are so many lines along which data and work could be effectively organized. I've debated this with some highly capable designer friends. I believe all the examples we've come up with where style parted with content, it's a side-effect of other separations:

- **Who-Separation**, separating one person's work from another's (Sandy's text from Joe's fonts)
- **Pattern-Separation**, separating data that varies from data that doesn't (each product's properties from the colour scheme in which they're all listed, or a manual's text from its paper and PDA layout styles)

Broad as these are I'm sure there are many more. Within each, the distinctions are intricate, and very important to the task, and very hard to discuss in the abstract.

If Sandy Philosopher writes a visionary paper and Joe Designer makes it look like a hip magazine article, we have style and content separation. But the seminal insight is that Sandy is articulate and Joe is artful. What's purposefully separated is Sandy's work from Joe's. Sandy and Joe know this, and so should their site architects.

There are significant opportunities in how you present information to affect how it's understood.

In response to Tim Berners-Lee's admonition in an early SGML paper against prima donna authors, I can only say touché: "some authors feel that the act of communication includes the entire design of the document, and if this is done correctly the formatting is an essential part of authoring. They resist any attempts to change the representation used for their documents."

My aversion to a rule-of-thumb separating style from content comes partly from a belief that some style is content. Not style to attract attention, but style to inform. Now, there is much art in

attracting attention and I don't mean to belittle that power, but there is also art to communicating well in visual media. In Tufte's *Envisioning Information*, quite the opposite of separation occurs. Presentation makes sweet love to structure when conceiving an informative vision: "To envision information – and what bright and splendid visions can result – is to work at the intersection of image, word, number, art." This is design inextricable from authorship. It's style that cannot be dissected from content without bleeding away informative power.

Synchrony of meaning and expression is the future of web design.

Of course, web authorship today is severely constrained, hidebound by the small set of universal, reliable browser features. Realizing what you envision often proves impractical without withering compromise. Also, good design must react with fluidity to variation in browser brand and version, monitor settings, window dimensions and other user diversities. But these challenges are not so much met by separating style from content as by separating universal features from unique ones, another variation on pattern-separation. As soon as a style becomes universally supported, the need to separate it from content disappears.

A publisher may limit an author's options to control appearance, so books can be efficiently produced on paper, CD and web. But as the media evolve, so could an author's options. An IT department rigidly committed to separate style and content would not imagine authors should ever expect relief from stylistic impotence, and might vigorously oppose an author yearning to express rich visual structure. A more enlightened IT architect could separate instead what's common to all media from what's unique (pattern-separation), and therefore find it natural for authors to draw upon a growing set of common media capabilities. Or she may decide to accommodate one author's wish (who-separation) to influence the web version of his book.

A working designer may dismiss the distinctions I'm trying to make as semantic and irrelevant, and they would be right, and that is in fact close to my point: when getting down to work, the religion of separate style and content is forgotten, something else is practiced.

But the distinction has enormous implications for the progress of web standards and browser implementation. In the rise of e-civilization, it's not enough to educate designers to separate style from content. Nowhere near enough. (Any more than it's enough for boot camp to teach about living in mud. Boot camp must empower soldiers to kill enemies no one has seen yet.) Technology and training must empower designers to organize along specialized dichotomies and multichotomies. To create structure no one has imagined yet.

Is this why the promise of XML is so exciting? Yes I think so. Not because it separates style from content, but because it can separate all kinds of things, organize in all kinds of ways. XML could enable a great deal if it empowers designers to invent widely usable structure. It's designers who must do the real work of inventing new structure, because they know the real world.

The Hungarian Academic Publishing House used Advent's 3B2 SGML/XML publishing software in producing a series of bilingual **Hungarian dictionaries** for print, CD and web. The story contains no mention of style and content separation. The project organized along its own unique lines:

- "The Hungarian knowledge base has to be identical for various dictionaries, English, French, German." (pattern-separation)
- "Publication for print, CD and for the web should be possible from the same database." (pattern-separation)
- "The system must enable off-line editing for teleworkers." (who-separation)

- “Alphabetical processing of special accented characters and double letters of the highly sophisticated Hungarian language required non-trivial additional programming.” (blurring the line between style and content)

Say the designer for a kayak manufacturer invents a standard Kayak Expedition Markup Language (KEML) so any weekend kayaker can describe local experiences. In some places the challenge is white water, in others it's alligators and snakes, others the city police. Perhaps the world can quickly become one big kayak adventure map – while it simultaneously becomes many other things to many other people. Decentralization can be very powerful. Standards committees can't begin to accommodate all the arcana of human existence. Yet that's exactly what we expect of the web.

Accessibility: the Politics of Design

Article written by Alan Herrell on January 10, 2001

A little over a year ago I wrote an article here on the upcoming U.S. Accessibility Regulations for the web. Specifically on Section 508 of the WORKFORCE INVESTMENT ACT OF 1998, which required all United States Federal Agencies with websites to make those sites accessible to individuals with disabilities, within 24 months of enacting of this law.

The Regulations are finally out and the clock has started ticking again.

"But I don't work for the government," you cry! Probably true. But the impact of these regulations will extend far beyond the .gov domain. If you work for a University or other .edu that takes federal grants and have a website, surprise! It's gonna be your day in the barrel too.

One of the lesser known provisions of the The Americans with Disabilities Act (ADA) is the *"effective communication rule."*

In a Letter Dated September 9, 1996, to Sen. Tom Harkin, Deval L. Patrick, Assistant Attorney General, Civil Rights Division had this to say,

“The Americans with Disabilities Act (ADA) requires State and local governments and places of public accommodation to furnish appropriate auxiliary aids and services where necessary to ensure effective communication with individuals with disabilities, unless doing so would result in a fundamental alteration to the program or service or in an undue burden. 28 C.F.R. . 36.303; 28 C.F.R. . 35.160. Auxiliary aids include taped texts, Brailled materials, large print materials, and other methods of making visually delivered material available to people with visual impairments.

“Covered entities under the ADA are required to provide effective communication, regardless of whether they generally communicate through print media, audio media, or computerized media such as the Internet. Covered entities that use the Internet for communications regarding their programs, goods, or services must be prepared to offer those communications through accessible means as well.”

This letter suddenly expanded the rules to include State and local government websites. It also opened the door for interpretation of accessibility claims against commercial websites. If you have a .com that does business with the government, you may need to determine if these regulations apply to you. Lots of sites, lots of lawyers, connect the dots.

This is Good News. Some of you are currently unemployed. The implosion of the pure play dot.com websites, with the arrival of these regulations, has presented a golden opportunity for new jobs. A lot of new jobs. Before sending out those resumes, let's see what work the government has done.

Uncle Sam Needs You

The regulations gave the enforcement authority to the Department Of Justice. On April 2, 1999, the Attorney General issued a memorandum to the heads of all Federal agencies advising them of the requirements of Section 508 and providing instructions for conducting self-evaluations of their electronic and information technology.

The United States Government has over 20,000 websites. 3,028 Web pages were surveyed.

This effort on the part of the DOJ, although comprising only a small sample, represented a good faith effort to measure compliance and to give a heads up on these regulations.

This information was collected, collated and presented in a report. On July 22, 1999 the Justice Department released this report on the state of Federal Websites. This report pointed out that a number of tools and technologies currently in use for disseminating information on the taxpayers dime are not accessible.

Note 4 of this report states:

"4. The Department was careful to limit the degree and scope of conclusions drawn from the data provided by agencies, for the simple reason that many of the components appeared to misunderstand some of the questions. Spot-checks conducted by the Department of the Web sites – the URL's of which were reported on the survey forms – revealed that many of them did not contain the features the components identified them as containing. For instance, 592 Web pages were identified as containing 'applets.' The Department, after reviewing a majority of these pages, did not find a single applet in a spot-check of most of them.

"There are many possible explanations for this observation. It is possible that as components identified accessibility problems with certain types of features (e.g., applets), they deleted the offending features from their Web pages rather than making them accessible. It is more likely, however, that many of those who evaluated Web pages were not sufficiently careful or knowledgeable to correctly identify features of their Web pages."

Taken to task in the presentation arena were Java applets, imagemaps, Microsoft PowerPoint and Adobe Acrobat .pdf files.

While robust, Java applets have a number of issues which, in order to present accessible information, makes their use problematic. Imagemaps with the correct use of alt-tags present less of a challenge, but require thought. PowerPoint presentations are severely limited by not only accessibility concerns, but also by requiring a copy of PowerPoint to view.

Adobe is one of the major providers of tools that are used daily across the world to build websites. Adobe's Portable Document Format is a flourishing document format that gives What-You-See-Is-What-You-Get or "WYSIWYG" real meaning. It is a very good format for presenting information considered final, which is why it used extensively for legal documents. That's the good news.

On July 22, 1999, the Department of Education issued an overall agency report. This report summarized the accessibility challenges faced by agencies which choose to put documents in Adobe Acrobat's pdf format:

The Portable Document Format (PDF) has provided one of the most controversial accessibility problems of the decade. PDF documents, by the nature of the medium, are portable, cross-platform, generally tamper-proof, and render in exacting detail, representations of the original print document's fonts, formatting, etc.

According to the Justice Department report:

"17. Recently, in a presentation to federal agency officials and employees, representatives from Adobe explained that their newly released version of Adobe Acrobat included many accessibility features that, if used correctly, could be used to create files that were easily accessible to users with disabilities. Adobe's Accessibility Seminar, Feb. 2, 2000, IRS Building. Word processed documents that are 'printed' to pdf instead of 'scanned' to pdf are much more likely to work well with Adobe's access utility. *Adobe clarified that it sees its job as simply to provide the tools for making accessible files, not to teach users how to use these tools to make files more accessible.*" [My italics]

Information Technology and People with Disabilities: The Current State of Federal Accessibility April 2000

The government has done its homework.

A private report came out in September: Assessing E-Government: The Internet, Democracy, and Service Delivery by State and Federal Governments. Here is the executive summary:

- 1) only 5 percent of government websites show some form of security policy and 7 percent have a privacy policy
- 2) 15 percent of government websites offer some form of disability access, such as TTY (Text Telephone) or TDD (Telephone Device for the Deaf) or are approved by disability organizations.
- 3) 4 percent offer foreign language translation features on their websites
- 4) 22 percent of government websites offer at least one online service
- 5) a few of the sites are starting to offer commercial advertising, which raises problematic issues for the public sector
- 6) 91 percent of the sites responded to a sample email requesting the official office hours of the particular agency and three-quarters did so within one business day
- 7) states vary enormously in their overall ranking based on our analysis. Texas, Minnesota, New York, Pennsylvania, and Illinois ranked highly, while Rhode Island, Delaware, New Hampshire, South Dakota, and Nevada did poorly
- 8) the best predictor of state rank was population size. Small states had access to fewer resources and had difficulty achieving economies of scale necessary for technology initiatives
- 9) in terms of federal agencies, top-rated websites included those by the Consumer Product Safety Commission, Department of Treasury, Department of Agriculture, Department of Education, and Federal Communication Commission. Poorly ranked agency websites included the National Security Council, U.S. Trade Representative, White House, U.S. Postal Service, and Thomas (the joint congressional website)
- 10) in general, federal government websites did a better job of offering information and services to citizens than did state government websites

- 11) judicial websites ranked more poorly on providing contact information than did executive or legislative sites
- 12) there is a need for more consistent and standard designs across government websites.

There is a lot of work to be done.

The accessibility regulations cover not only the methods for creating accessible websites, but also the purchase of the authoring tools for creating and managing content. Stay tuned for new versions of your favourite tools with accessibility features.

The Developers' Dilemma

There are over 20 million registered domain names, around five billion pages, and another 1000 sites have come online since you started reading this article.

The Centre for Applied Special Technologies, whose BOBBY Validator allows you to check your sites for Accessibility, has a page devoted to sites that validate for Accessibility. There are only 1284 entries as of this writing.

The code we use is changing. HTML 4.01, XML, CSS2, all of these are getting better for us, but as we wait for the browsers to catch up, we also must remember that not everyone has our toys, is able-bodied, or even sees.

The range of tools for coding spans the spectrum from text editors to visual design suites that create code as you move design elements across your screen. Which in some cases is a magnificent collection of the finest hardware and software available. But the best tools on the market will not guarantee accessible code or design. This is your job.

The tools for enabling accessibility are free. That's right!! F R E E. The W3C Validator is free. The W3C CSS Validator is Free. The BOBBY Validator is Free.

Welcome to browser hell. There are three major visual browsers, Internet Explorer, Netscape Navigator, and Opera. None of these allow you to use the entire range of accessibility tags. Nor is the support for the usable ones consistent. Platform dependencies create rendering issues. Lynx is a text browser. Limited formatting and no visuals allowed here. Screen Readers require a whole new mindset in respect to using tables for presenting content.

Testing, Testing, Testing. You can build the cleanest pages possible and validate them, but unless you test them in every browser you can get your hands on and get your friends who work in different environments to test them, you will not be sure.

If you build websites for business, sooner or later the folks whose money you took will probably get email from someone who wants to buy your client's stuff, but can't because you made a decision not to make their site accessible.

These regulations point to a new era of website development. Accessibility in website design is not a crisis of confidence, it is a challenge to your creativity, knowledge, and ability to create a web for everyone. Governments will do it, because it is our tax dollar. Companies will do it because it is a bottom line issue. You will do it because you love a challenge

Lukáš Zdechovan