



THE JOURNAL ON
TECHNOLOGY AND
PERSONS WITH
DISABILITIES

Meeting Accessibility Challenges with Web Components

Jason White, Mark Hakkinen, Jennifer Grant

Educational Testing Service

jjwhite@ets.org, mhakkinen@ets.org, jgrant@ets.org

Abstract

Web Components comprise a suite of technologies under development by the World Wide Web Consortium (W3C) that together enable standard formats such as Hypertext Markup Language (HTML), and Scalable Vector Graphics (SVG) to be extended with new functionality. Web Components encapsulate presentation and behavior in a reusable fashion that coheres well with the markup languages and development practices of the Web. We briefly review the constituents of Web Component technology - custom elements, the shadow DOM, HTML imports, and the HTML 5 template element. We then argue that Web Components are a general mechanism that can be used to address problems, some of them long-standing, of Web accessibility.

Our argument proceeds via a series of examples that illustrate the utility of Web Components as means of enhancing non-visual access to images, providing spoken and braille alternatives to textual content, and implementing custom interactive controls with features that improve access while bringing to the fore the underlying semantics of the content. Although the design of our Web Components is motivated by educational needs, they and the broader approach to solving accessibility challenges which they exemplify are applicable to the Web as a whole.

Keywords

Web accessibility, Web components, Web standards.

Introduction

Although the Web browser has become a progressively more sophisticated programming environment, until recently, its predominant markup languages - HTML and SVG - have not been accompanied by an extension mechanism permitting new elements and attributes to be defined to express application or domain-specific concepts. This role is now filled by Web Components, a technology which is presently undergoing standardization by the World Wide Web Consortium, and which browser developers have started to implement.

Web Components support abstraction and modularity in Web applications by allowing the registration of "custom elements" (Glazkov, *Custom Elements*) that can be used in HTML or SVG contexts. A custom element is typically associated with executable code (written in JavaScript) with which the element provides a user interface. This user interface can be implemented via HTML, SVG and CSS. The code and styles comprising the user interface operate in isolation from scripts and CSS rules applicable to the HTML document in which the custom element occurs. This separation is achieved by way of the shadow DOM (Glazkov and Ito), a mechanism for attaching an element hierarchy to a DOM node. This subordinate tree is then isolated from the surrounding document.

A simple mechanism, HTML imports (Glazkov and Morrita) has been proposed to allow Web Component implementations to be included in HTML documents. This proposal, which extends the HTML link element, remains controversial, and it may be superseded by an alternative solution. Authors of Web Components can also take advantage of the template element, already standardized in HTML 5, (Hickson et al. 4.11.3) to supply prebuilt content for manipulation by JavaScript code in constructing the desired user interface. In combination, the technologies briefly introduced here support a high degree of modularity. A custom element, with its associated functionality and styles, comprise a Web Component which, once created, can be used as needed in diverse application scenarios while maintaining isolation from the surrounding context.

As we shall demonstrate via the examples described in the next section of the paper, the emergence of Web Components opens the possibility of devising reusable solutions to problems of Web accessibility that can be integrated into documents and applications in a straightforward and modular fashion. Web Component technology provides markup language extensibility of a kind that complements and enhances the capabilities of the Web browser as a programming

environment. The potential benefits that this extensibility brings to addressing the challenge of accessibility are, we contend, considerable. This conclusion is here supported by experiments in the design of innovative Web components that address a range of practical difficulties which current Web standards do not adequately resolve: the provision of alternatives to images that extend beyond textual descriptions, the specification of precise spoken and braille representations of text, and the creation of custom user interface components.

Web Components for Associating Alternatives with Images

Standards and practices for making graphical content accessible have focused largely on the use of textual alternatives such as labels and descriptions. This emphasis is reflected in guideline 1.1 of *Web Content Accessibility Guidelines (WCAG) 2.0* (Caldwell et al.), which requires text alternatives to be provided for any "non-text content", including images. Although this requirement serves the needs of users of conventional screen readers, it does not accommodate a larger range of non-visual representations that can provide richer perceptual alternatives to graphical material. Moreover, the needs of users with learning and cognitive disabilities are not taken into consideration, due to a lack of universally applicable measures that are widely accepted as enhancing such users' ability to comprehend graphical content. A more inclusive solution, the Diagrammar, has recently been proposed by the DAISY Consortium (*Resource Directory for the Z39.98-2012 Authoring and Interchange DIAGRAM Description Feature, Version 1.0*) as an XML format designed to facilitate the authoring and exchange of a range of alternatives that may be associated with an image. The supported alternatives include a short and a long description, a tactile graphic (supplied as a reference to an external resource), a tour (i.e., a textual overview) of the tactile graphic, a simplified image and a simplified description. Whereas the last two alternatives are meant to meet the needs of people with learning and cognitive disabilities, the remainder are principally of benefit to users who are blind or who have low vision.

By means of Web Components, the Diagrammar can be implemented directly in HTML as a set of custom elements, preserving the original semantics and making only slight syntactic adjustments. Specifically, the specification requires each custom element to be given a name that contains a hyphen ("-") character, and which avoids names reserved by other specifications. An instance of our dg-content Web components is depicted in figure 1. Unlike the original XML

format intended to enable the creation and exchange of documents, a series of Web Components is expected to provide not only syntax but also behavior. In adapting the Diagrammar for use in Web applications, we chose an approach to the design which is inspired by the audio and video elements of HTML 5. If the Boolean controls attribute is set, then the Web Components display a user interface, namely a menu of alternatives to the original image (illustrated in figure 2) from which the user can select. If the controls attribute is not set, the determination of which alternatives to present to the user is the responsibility of the application in which the Web Components occur. This flexibility supports the use of a global configuration option or a profile of the user's access requirements to control the selection of alternatives. Thus, the Web Components are designed to be consistent with the recent paradigm shift in accessibility research and standards toward personalization of user interfaces according to each individual's declared needs and preferences (Nevile; Nevile, Treviranus, and others; Vanderheiden et al.).

```

<dg-content controls>
  <dg-img>
    
  </dg-img>
  <dg-summary show>
    The image depicts two surveyors measuring the angles between
    themselves and a tree.
  </dg-summary>
  <dg-longdesc show="true" overlay="false">Two surveyors, A and P, stand some distance
  apart on the south bank
  of a river, looking at a tree, T, that is on the north bank of the river.
  Points A, P, and T form a triangle. At points A and P, there are two parallel
  sight lines pointing north and forming angles outside of the triangle. At
  point P, angle TPA is 53 degrees. The adjacent angle between PT and the
  northern sight line is 37 degrees. At point A, angle TAP is not labeled,
  and the adjacent angle formed between AT and that northern sight line is
  32 degrees.
  </dg-longdesc>
  <dg-simplifieddesc show="false">
    T, A, and P are the three points on a triangle. Angle TPA is 53 degrees
    with an adjacent angle of 37 degrees. The angle adjacent to angle TAP is 32 degrees.
  </dg-simplifieddesc>
  <dg-tactile show="false" source="imgs/anglesmap.jpg" controls="false">
    <dg-tour>
      Start exploring at the top right.
    </dg-tour>
  </dg-tactile>
</dg-content>

```

Fig. 1. Diagrammar implemented as a web component. The code can be used in line within an HTML application.

DIAGRAM Content Model in HTML

▼ Controlling <dg-content> with javascript

Show/Hide Controls

Show/Hide longdesc

Show/Hide Simplified Description

Show/Hide Tactile

Toggle longdesc style

Survey Diagram

The following image:

Description

The image depicts two surveyors measuring the angles between themselves and a tree.

Two surveyors, A and P, stand some distance apart on the south bank of a river, looking at a tree, T, that is on the north bank of the river. Points A, P, and T form a triangle. At points A and P, there are two parallel sight lines pointing north and forming angles outside of the triangle. At point P, angle TPA is 53 degrees. The adjacent angle between PT and the northern sight line is 37 degrees. At point A, angle TAP is not labeled, and the adjacent angle formed between AT and that northern sight line is 32 degrees.

Simplified: T, A, and P are the three points on a triangle. Angle TPA is 53 degrees with an adjacent angle of 37 degrees. The angle adjacent to angle TAP is 32 degrees.

Emboss

Tour: Start exploring at the top right.

Can you explain how the distances to T is calculated?

Fig. 2. Diagrammar Content from Figure 1, as rendered in HTML using the Chrome Browser.

Providing Spoken and Braille Alternatives to Text

Our second example demonstrates the use of Web Components to meet an immediate need arising from shortcomings in the implementation of standards by assistive technologies. Since current screen readers do not support Speech Synthesis Markup Language (SSML) (Burnett, Shuang, and others), there is no standards-based mechanism available with which a content author can prescribe the pronunciation of a text string, for example a name or a domain-specific term. Although labeling techniques (e.g., the `aria-label` attribute) can be used as a crude means of substituting a misspelling of a word or an expansion of an abbreviation that improves pronunciation, they incur the disadvantage of imposing the substitution not only on text to speech users, but also, and undesirably, on users of refreshable braille displays. The comprehension of text to speech users is thus improved at the expense of the comprehension of braille users, for whom the misspelling of key terms, for example, runs the risk of impeding understanding of the text.

Until SSML is more widely deployed, this problem can be partially addressed by a set of Web Components, comprising a container, `apip-alt`, which allows spoken and braille renderings to be specified in their respective child elements (see figure 3). The name is a reference to the Accessible Portable Item Protocol (APIP) specification of the IMS Global Learning Consortium (Accessible Portable Item Protocol (APIP)). The spoken form is a string that can be substituted for the original text; the braille form is given as a string of characters drawn from a block of Unicode code points that correspond to all 256 possible eight-dot braille patterns. In practice, of course, six-dot braille cells would typically be used, as these are the norm in almost all braille encodings. This braille representation takes advantage of the observation that the Unicode braille characters are correctly processed by most screen readers (i.e., the corresponding dot patterns are forwarded to the braille display). As in the preceding example, a `controls` attribute determines whether the Web Component supplies its own user interface or whether this responsibility belongs to the application. In the resultant rendering shown in figure 4, the control is rendered as a drop down list prior to the displayed text, which enables the user to select their preferred alternate from the list (braille or spoken). The effect of the Web Component is thus to present either the spoken alternative or the braille alternative to the user, while allowing for the

possibility of switching between them, thereby supporting the needs of those who rely on both access methods during a single interaction.

```

<h2>Sample Problem</h2>
<p>The following is a math expression:</p>

<apip-alts id="s0" controls="true">
  <apip-display>12<var>x</var><sup>2</sup>
    + 7<var>xy</var> - 10<var>y</var><sup>2</sup>
</apip-display>
  <apip-spoken id="s1" >
    Twelve times ex squared, plus seven times ex times
    why, minus ten times why squared
  </apip-spoken>
  <apip-braille id="s2" >
    #12x^2"+7xy-10y^2_4
  </apip-braille>
</apip-alts>

<p>Can you factor this expression?</p>

```

Fig. 3. Prototype apip-alts implemented as a web component. The code can be used in line within an HTML application. The element contains "display text", spoken alternate, and Nemeth versions of a simple equation.

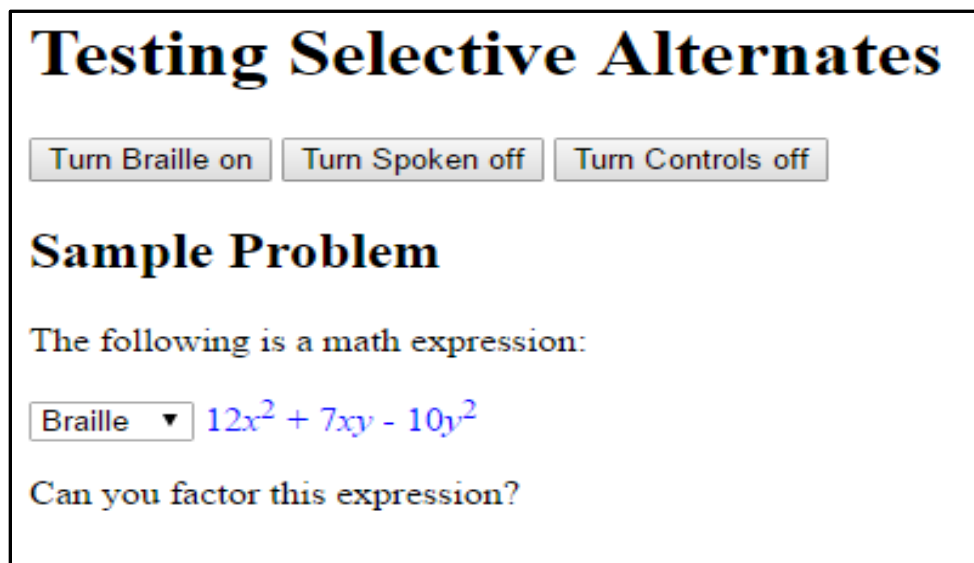


Fig. 4. apip-alts content from Figure 3, as rendered in HTML using the Chrome Browser.

Designing an Interactive Control as an Accessible Web Component

The third example was developed to demonstrate the value of Web Components in capturing the semantics of complex user interface controls, while providing intrinsic accessibility by implementing the WAI-ARIA 1.0 attributes required to support screen readers (Craig and Cooper). A set of Web Components was developed that implements the capabilities of the choice element defined in the IMS Global Learning Consortium Question and Test Interoperability (QTI) specification, an XML-based standard for the creation and exchange of interactive assessment items (*IMS Question & Test Interoperability Specification*). The choice element supports both single and multiple selection of options by the user, as well as the automatic shuffling of the choices. To the user, the resulting qti-choice Web Component (see figure 5) and its contents appear as a series of radio buttons (see figure 6), with customary behavior. The advantage to the assessment application author, however, lies in the provision of a modular component that can easily be incorporated into software for delivering tests to users via a Web browser, or even into the test materials themselves, while maintaining the supplied visual presentation, behavior, and accessibility, and preserving the abstract semantics of the original XML format. The visual presentation can readily be modified by application authors who desire to do so.

```

<qti-itemBody>
  <p>Look at the picture of a sign found in Japan.</p>
  <p></p>


  <qti-choiceInteraction responseIdentifier="RESPONSE" shuffle="false" maxChoices="0"
    minChoices="0" orientation="vertical">
    <qti-prompt>What does it indicate to you?</qti-prompt>
    <qti-simpleChoice identifier="ChoiceA">You have reached a safety point.</qti-simpleChoice>
    <qti-simpleChoice identifier="ChoiceB">Watch out for open manhole covers.</qti-simpleChoice>
    <qti-simpleChoice identifier="ChoiceC">Watch out for puddles of water.</qti-simpleChoice>
    <qti-simpleChoice identifier="ChoiceD" fixed="true">None of the above.</qti-simpleChoice>
  </qti-choiceInteraction>
  <qti-submit></qti-submit>
</qti-itemBody>

```

Fig. 5. Prototype QTI Choice Interaction implemented as a web component. The code can be used in line within an HTML application.

► Change Options

Look at the picture of a sign found in Japan.



What does it indicate to you?

You have reached a safety point.

Watch out for open manhole covers.

Watch out for puddles of water.

None of the above.

Submit Response

Fig. 6. apip-alts content from Figure 5, as rendered in HTML using the Chrome Browser.

Discussion

As the preceding examples illustrate, Web Components permit the HTML and SVG markup languages to be extended without requiring any modification of the underlying browser's source code, hitherto the only effective means of associating novel behavior with newly defined elements. The custom elements can be applied without having regard to the details of their implementation: presentation and behavior, including characteristics needed to support accessibility, are safely isolated within the Components themselves. A familiar and expressive markup language interface - that of elements and attributes - is presented to the content author, who is insulated from the internal details of the Web Components, including, crucially, those

aspects of their implementation that are needed in order to support accessibility. The semantics of the content are clearly exhibited in the markup by way of appropriate abstractions, a practice made possible by Web Components that may be contrasted with the repurposing of HTML elements, including generic containers such as `div` and `span`, as user interface controls, in which the true meaning of the markup is obscured from authors and maintainers of the document or application. Thinking and designing in terms of suitable abstractions and their associated vocabulary is thus encouraged by the introduction of special-purpose elements as Web Components.

Conclusion

As our examples show, Web Components provide a versatile, expressive and modular means of bringing abstractions into Web applications, with accompanying user interfaces that address problems of accessibility. Ongoing work by the authors and their collaborators seeks to integrate the "Diagrammar" Web Components into a reading system for electronic books, and to inform the development of future versions of the IMS QTI standard by encouraging support for the use of Web Components as a modular and accessible means of building interactive assessments for delivery via Web technologies.

Our examples explore only a few of the potential ways in which Web Components can be applied to enhancing accessibility. As support for the underlying technologies matures, it will become increasingly feasible to develop and deploy in practice a host of further innovations based on Web Components designed to meet the needs of users with disabilities, and to assist authors in the design of accessible Web applications.

Works Cited

- Accessible Portable Item Protocol (APIP)*. IMS Global Learning Consortium, 2016. Web. <https://www.imsglobal.org/apip/index.html>.
- Burnett, Daniel C., Zhi Wei Shuang, and others. *Speech Synthesis Markup Language (SSML) Version 1.1*. W3C, 2010. Web. <http://www.w3.org/TR/speech-synthesis11/>.
- Caldwell, Ben et al. *Web Content Accessibility Guidelines (WCAG) 2.0*. W3C, 2008. Web. <http://www.w3.org/TR/wcag20/>.
- Craig, James, and Michael Cooper. *Accessible Rich Internet Applications (WAI-ARIA) 1.0*. W3C, 2014. Web. <http://www.w3.org/TR/2014/REC-wai-aria-20140320/>.
- Glazkov, Dimitri. *Custom Elements*. W3C, 2016. Web. <http://www.w3.org/TR/2016/WD-custom-elements-20160226/>.
- Glazkov, Dimitri, and Hajime Morrita. *HTML Imports*. W3C, 2016. Web. <http://www.w3.org/TR/2016/WD-html-imports-20160225/>.
- Glazkov, Dimitri, and Hayato Ito. *Shadow DOM*. W3C, 2015. Web. <http://www.w3.org/TR/2015/WD-shadow-dom-20151215/>.
- Hickson, Ian et al. *HTML5: A Vocabulary and Associated APIs for HTML and XHTML*. W3C, 2014. Web. <http://www.w3.org/TR/2014/REC-html5-20141028/>.
- IMS Question & Test Interoperability Specification*. IMS Global Learning Consortium, 2016. Web. <https://www.imsglobal.org/question/index.html>.
- Nevile, Liddy. "Adaptability and Accessibility: A New Framework." *Proceedings of the 17th Australia Conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*. Computer-Human Interaction Special Interest Group (CHISIG) of Australia, 2005. 1–10. Print.
- Nevile, Liddy, Jutta Treviranus, and others. "Interoperability for Individual Learner Centered Accessibility for Web-Based Educational Systems." *Educational Technology & Society* 9.4 (2006): 215–227. Print.

Resource Directory for the Z39.98-2012 Authoring and Interchange DIAGRAM Description Feature, Version 1.0. DAISY Consortium, 2014. Web.

<http://www.daisy.org/z3998/2012/auth/features/description/1.0/>.

Vanderheiden, Gregg C et al. “Auto-Personalization: Theory, Practice and Cross-Platform Implementation.” *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 56. SAGE Publications, 2012. 926–930. Print. 1.

Journal on Technology and Persons with Disabilities

ISSN 2330-4216

LIBRARY OF CONGRESS * U.S. ISSN CENTER
ISSN Publisher Liaison Section
Library of Congress
101 Independence Avenue SE
Washington, DC 20540-4284
(202) 707-6452 (voice); (202) 707-6333 (fax)
issn@loc.gov (email); www.loc.gov/issn (web page)

© 2016 The authors and California State University, Northridge

This work is licensed under the Creative Commons Attribution-NoDerivs 4.0 Unported License.

To view a copy of this license, visit <http://creativecommons.org/licenses/by-nd/4.0/>

All rights reserved.

