2015

www.g3ict.org
G3ict Publications & Reports

Business Case White Paper Series

PhoneGap/Cordova: A key to apps portability for all users

Global Initiative for Inclusive Information and Communication Technologies

Researched in cooperation with Adobe Systems, Inc.

PhoneGap/Cordova: A key to apps portability for all users

A G3ict Business Case White Paper Series:
Researched in cooperation with Adobe Systems, Inc.

February 2015

About G3ict

G3ict is an Advocacy Initiative of the United Nations Global Alliance for ICT and Development, launched in December 2006, in cooperation with the Secretariat for the Convention on the Rights of Persons with Disabilities at UN DESA.

Its mission is to facilitate and support the implementation of the dispositions of the Convention on the Rights of Persons with Disabilities in promoting e-accessibility and assistive technologies. G3ict participants include industry, the public sector, academia and organizations representing persons with disabilities. G3ict relies on an international network of ICT accessibility experts to develop practical tools, evaluation methods and benchmarks for States Parties and Disabled Persons Organizations to implement policies in support of assistive technologies and e-accessibility.

Since inception, G3ict has organized or contributed to more than 90 awareness-raising and capacity-building programs for policy makers in cooperation with international organizations such as the ITU, UNESCO, UNITAR and the World Bank.

G3ict co-produces with ITU the "e-Accessibility Policy Toolkit for Persons with Disabilities" (www.e-accessibilitytolkit.org), which is widely used around the world by policy makers involved in the implementation of the Convention on the Rights of Persons with Disabilities. For additional information on G3ict, visit www.g3ict.org

G3ict White Paper Business-Case Series

The G3ict White Paper Business-Case Series documents innovative accessibility solutions and good practices with real-world case studies for users and organizations seeking to improve the accessibility of their information technology, applications and services.

Acknowledgments

G3ict wishes to express its sincere appreciation to Adobe Systems, Inc. for supporting G3ict activities and to Matt May, Accessibility Evangelist, Michael Jordan, Accessibility Engineer, and Kiran Kaja, Accessibility Program Manager, Adobe Systems, Inc., whose help in documenting this White Paper has been invaluable.

Sincere thanks also to: Christian Allen, CTO Crumbdrop, Victor Tsaran, Technical Program Manager, Google, Mubasshir Pawle, Chief Operations Officer, Eiosys, Inc., Erik, Ictenablers, for their comments and responses to our inquiries.

Editorial Team

Christine Forget-Leblois, G3ict Editor
Francesca Cesa Bianchi, Reviewer
Design by Manuel Ortiz - www.modesignstudio.com

CONTENTS

Introduction

As smartphones become more prevalent and more capable, the market for mobile apps is becoming more diverse and demanding than ever. Users of mobile devices have come to expect a level of support approaching if not equaling what they would receive on a desktop application or web browser. These apps have gone from toys and curiosities to critical points of contact between people and their workplaces, schools and governments.

Now that mobile apps have reached critical mass, the same level of detail is necessary to ensure that everyone has an equal opportunity to use them effectively. While just a decade ago the concept of accessibility to people with disabilities meant supporting a small handful of screen readers and magnifiers on Windows, the ecosystem for assistive technology in 2014 is exceedingly diverse. The top four shipping mobile operating systems each offer their own set of tools ranging from screen readers and magnifiers to support for refreshable Braille displays, high-contrast modes, and voice recognition. These features can allow mainstream applications to be highly accessible to mobile users.

Critically, however, each operating system offers its own application programming interface (API) to developers, meaning that app developers need to learn and implement a new API for each platform they target. For any cross-platform application, this is a significant barrier. But for accessibility, this process is exponentially harder. Developers need to take into account not only the user interface conventions of each platform, but also the level of accessibility support for each operating system, and even the order in which each feature was introduced, then develop custom, non-reusable code for each app. Predictably, the end result has been limited, inconsistent or even non-existent accessibility support for cross-platform mobile apps.

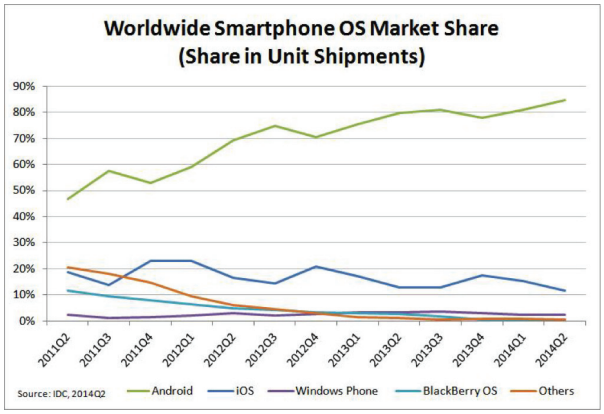
As a consequence, the adoption of mobile devices by users with accessibility issues, including seniors, has been slower than could be expected. Furthermore, lack of such features can affect compliance with existing rules and regulations regarding accessibility (including the American Disabilities Act, the Communications and Video Accessibility Act – CVAA – and the UN Convention on the Rights of Persons with Disabilities).

Adobe PhoneGap was created to simplify mobile development by using each platform's web browser component as a programming interface, allowing developers skilled in HTML, CSS and JavaScript to build apps that can be compiled to run on any given operating system. Adobe Accessibility, in collaboration with the PhoneGap team, was able to extend this write-once, run-everywhere strategy to mobile accessibility through a new module that gives developers access to each device's built-in accessibility features. App developers are now able to use the web technologies they already know, like the WAI-ARIA API, to build apps that take advantage of each platform's accessibility support to the greatest extent possible.

PhoneGap itself is open-source software, published by the Apache Foundation under the name Apache Cordova. This made it possible to add this sweeping new functionality quickly, test it within the accessibility community, and deploy and update the module without waiting to publish on PhoneGap's release cycle. This same process will allow us to add new accessibility features as each operating system evolves and improves. With this new functionality, we are able to help improve the accessibility of tens of thousands of apps built using PhoneGap technology.

Mobile devices and apps: so many choices, so many obstacles

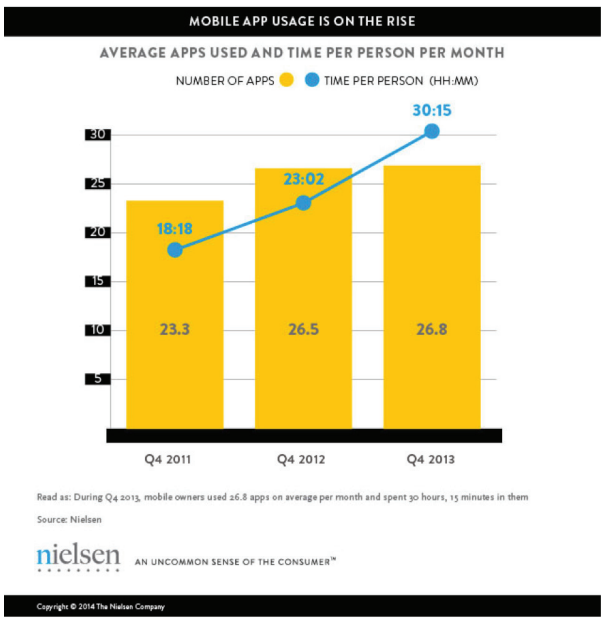
"There's an app for this": this is the new mantra of mobile devices users. As adoption of mobile technologies grows, so does the market for applications.



Mobile devices and connections are delivered worldwide at a pace of some half a billion units a year, with smartphones representing three quarters of the annual growth, and smart mobile devices in general 21% of the total market in 2013. At this rate, by the end of 2014, the number of mobile-connected devices will have exceeded the number of people on earth, and by 2018 there will be nearly 1.4 mobile devices per capita. There will be over 10 billion mobile-connected devices by 2018, including machine-to-machine (M2M) modules—exceeding the world's population at that time (7.6 billion) (Source: CISCO Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018).

Presently, several operating systems share the market, with the lion's share going to Android on the global market, though iOS Apple is the leader in the US market.

Each operating system has its own family of tens of thousands of applications, and the trend is continuing as users spend more and more time - and money - on apps, according to the most recent Nielsen study on adults 18 and older: (Android and iOS), concluding that "while there may be an upper limit to the total number of apps users are willing to access within a given month, the amount of time they are spending on those apps is showing no signs of slowing down".



Smart mobile devices are now the fastest growing source of data traffic, and global mobile data traffic grew 81 percent in 2013. Last year's mobile data traffic was nearly 18 times the size of the entire global Internet in 2000 (CISCO Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018). Social and search applications lead the usage:

TOP 10 SMARTPHONE APP CATEGORIES BY TIME PER PERSON

CATEGORY	Q4 2012	Q4 2013	GROWTH	% GROWTH
TOTAL	23:02	30:15	7:13	31%
SEARCH, PORTALS, & SOCIAL	8:33	10:56	2:22	28%
ENTERTAINMENT	6:11	10:34	4:22	71%
COMMUNICATION	3:29	3:48	0:19	9%
PRODUCTIVITY & TOOLS	2:00	2:16	0:16	14%
COMMERCE & SHOPPING	1:23	1:33	0:09	12%
NEWS & INFORMATION	1:00	1:33	0:33	55%
TRAVEL	1:14	1:18	0:04	6%
FAMILY & LIFESTYLE	0:59	1:16	0:16	29%
PHOTOGRAPHY	0:26	1:01	0:34	131%
FINANCE	0:33	0:35	0:03	10%

Read as: During Q4 2013, smartphone owners spent nearly 11 hours per person, on average, accessing social and search apps.

Source: Nielsen

nielsen AN UNCOMMON SENSE OF THE CONSUMER™

What does this mean for apps developers: they have to adjust to a permanently evolving market, and address specific technical definitions depending on the operating system they want to place their app on. Every app therefore has to develop several versions, a time and money consuming constraint, to reach the maximum possible number of users.

Also, as adoption of smart mobile devices becomes more mainstream, new groups of users appear: while seniors were not early adopters of smart phones, they now have joined the trend, and have also enthusiastically adopted tablets and e-readers:

Smartphone owners in 2014

Among adults, the % who have a smartphone

		Have a smartphone phone
	All adults	**58%**
	Sex	
a	Men	61
b	Women	57
	Race/ethnicity*	
a	White	53
b	African-American	59
c	Hispanic	61[a]
	Age group	
a	18-29	83[bcd]
b	30-49	74[cd]
c	50-64	49[d]
d	65+	19
	Education level	
a	High school grad or less	44
b	Some college	67[a]
c	College+	71[a]
	Household income	
a	Less than $30,000/yr	47
b	$30,000-$49,999	53
c	$50,000-$74,999	61[a]
d	$75,000+	81[abc]
	Community type	
a	Urban	64[c]
b	Suburban	60[c]
c	Rural	43

Source, Pew Research Center Internet Project Survey, January 9-12, 2014. N=1,006 adults.
Note: Percentages marked with a superscript letter (e.g., [a]) indicate a statistically significant difference between that row and the row designated by that superscript letter, among categories of each demographic characteristic (e.g., age).
* The results for race/ethnicity are based off a combined sample from two weekly omnibus surveys, January 9-12 and January 23-26, 2014. The combined total n for these surveys was 2,008; n=1,421 for whites, n=197 for African-Americans, and n=236 for Hispanics.

PEW RESEARCH CENTER

Who owns tablets and e-readers?

Among all American adults ages 18+, the % who own either a tablet computer or an e-reader

		Tablet	E-reader
	Total (All adults 18+)	**42%**	**32%**
	Sex		
a	Male	42	29
b	Female	43	33
	Race/ethnicity		
a	White	41	35[bc]
b	Black	34	24
c	Hispanic	45	18
	Age group		
a	18-29	48[d]	28
b	30-49	52[cd]	40[ad]
c	50-64	37[d]	32[d]
d	65+	25	22
	Education level		
a	High school grad or less	29	22
b	Some college	45[a]	33[a]
c	College graduate	59[ab]	44[ab]
	Household income		
a	< $30,000	26	14
b	$30,000-$49,999	45[a]	36[a]
c	$50,000-$74,999	47[a]	42[a]
d	$75,000+	65[abc]	53[ab]
	Community type		
a	Urban	43	32
b	Suburban	43	32
c	Rural	38	29

Source: Pew Research Center's Internet Project Omnibus Survey, January 2-5, 2014. N= 1005 American adults ages 18 and older. Interviews were conducted on landlines and cell phones, in English and Spanish.

PEW RESEARCH CENTER

Web View for any platform

Every mobile device needs access to the web simply to be marketable. While many different browser engines can run on any given operating system, each OS has its own deeply-integrated browser: for example, Mobile Safari on iOS, or Internet Explorer on Windows Phone. (Android's older browser, simply called Browser, was replaced in its KitKat release with a version of Chrome.)

Each of these mobile browsers is actually a user interface around a core set of web libraries known colloquially as a "web view." And each OS allows developers to build the native web view into an app, as a way to provide a consistent appearance for web content across all apps on a device. Its usage can range from a single panel to entire apps to the operating system itself. As its use becomes pervasive throughout a system, the need for OS vendors to build accessibility into web views eclipses that of any other component, which means a web view is the most reliable way to make complex content directly accessible on any given platform.

These web views are roughly equivalent in functionality, though they are written in entirely different languages, come from different engines, have different levels of support for experimental features, and can process HTML, CSS and JavaScript in very different ways.

Still, despite these conflicts, web views are meant to do one thing: display web content. It's a skill more companies are likely to find in-house than any compiled language, and that makes app development approachable to a much larger number of designers and developers.

It's also much easier to find someone with a working understanding of web accessibility than software accessibility. People who've worked closely with accessibility APIs and assistive technology in software are extremely hard to come by because every piece of the puzzle is interdependent and sparsely documented. That hard-won experiential knowledge makes software accessibility specialists especially valuable.

Training people on web accessibility is substantially easier, in comparison. Web accessibility emphasizes features such as good document structure and textual alternatives for media elements, which translate well to someone with a good understanding of HTML. By relying on web views as the basis for the PhoneGap framework, users benefit from the work already done on the operating system's browser, allowing them to use the same skills they'd been using on the web.

Mobile OS	Browser	Engine	Language
Android	Chrome	Blink	Java
iOS	Safari	WebKit	Objective-C
Windows Phone	Internet Explorer	Trident	C++
Firefox OS	Firefox	Gecko	C++

The process for building an app in PhoneGap is virtually identical to building a mobile website, except for how it is packaged and deployed. One way of looking at an app written in PhoneGap is as a set of web pages wrapped in executable code. It will function like typical HTML code would on any given host device, including built-in accessibility support—with one exception.

For privacy reasons, web content isn't allowed access to assistive technology status. The application programming interfaces (APIs) simply don't exist. This is necessary on the web to keep sites from actively excluding people for their use of accessible technology. A mobile device, however, is a much more personalized sphere of control than the entire web. The user depends on a higher degree of customizability for their device and the apps they need.

So mobile operating systems have their own accessibility APIs available to developers to report whether a specific accessibility feature is engaged, so it can communicate what's happening on the device when it's running, and stay idle when it's not. (Battery life also explains the need for this status.)

Now that each operating system has its own accessibility support, and the feature sets roughly overlap, cross-platform development tools like PhoneGap were at a functional disadvantage to native toolkits in that they couldn't query or respond to changes in the accessibility status of a given device. That's the reason for the PhoneGap Mobile Accessibility plugin.

Christian Allen CTO Crumbdrop, USA

"In general, I think PhoneGap/Cordova is a very important technology, and one I recommend to almost all other startups and small software businesses. Currently, based on consumer popularity, there's a near-imperative case for distributing your apps via "mobile web", Android, and iOS platforms- at a minimum. All three of these platforms are not only disparate infrastructures- hardware and underlying operating system- but they simply behave differently and users have different expectations when using them.

Instrumenting a great user experience is really difficult business when you only have one platform to worry about, and it can be prohibitively expensive or catastrophic for a small organization trying to meet those demands across the 3 platforms aforementioned.

PhoneGap allows you to focus your energy on the user experience and less on duplicating and rebuilding your functionality in multiple languages for multiple platforms. Ultimately, it allows for better, more compliant, across-the-board accessibility- and improved usability in general. That isn't to say it isn't without its own set of hiccups and pitfalls- there is complexity and implementation details to overcome, and there are some things that simply can't be translated from mobile web to iOS/Android (for example, a Facebook "Like" button, believe it or not). And there is much work to be done to get PhoneGap and the various plugins you will need working in your build and deployment process. But in the end, the World Wide Web has always demanded we move toward unifying our app development process so that we can have excellent apps that work for everybody. Just as it killed the market for desktop-based applications in favor of website-based applications that work in ubiquitous "browsers", it will soon, too, kill native smartphone-based applications in favor of the same. Just with a smaller visual footprint. And Cordova, PhoneGap, Ionic, and the other unifying tools that are building on each other right now will expedite this inevitable change".

Accessibility features are available today for users of all abilities

In some respects, the experience of using a mobile device reflects what many people with disabilities face when using any kind of computer. The viewport can display a limited amount of information, which is a familiar problem to users of screen magnifiers. People with fine motor problems may find the same problems typing or tapping on a small target that any user may face in a car or on the train. And changes in light or noise can make any display hard to read. Designing with the knowledge that all users face constraints like these is an important step toward accessibility in the mobile world.

Some of these issues compound one another to make it extremely difficult for mobile users with disabilities. For example, a small, low-resolution display coupled with a bright sun and a person with an eye injury requires more than just good software design to overcome. It's for reasons like these that mobile devices require built-in accessibility support similar to what can be found on PCs: screen magnifiers, screen readers, flashing and vibration for audio cues, high-contrast modes, voice recognition, tools for ignoring stray taps and more have found their way into the mainstream mobile tools we use today.

In the desktop world, if you needed one or more of these tools, your options were normally either something that was free and rudimentary or a professional product that cost hundreds, if not over $1,000. Custom hardware solutions often reached several thousand dollars. Early attempts in the mobile space were similar: they offered custom apps for smartphones and feature phones at a cost often greater than the device itself, or custom mobile devices offering features like Braille support at a cost in the thousands of dollars.

The modern mobile landscape, by contrast, offers a wide array of reliable accessibility support at no cost, including support for the user's existing hearing aid or external Braille display. To name but a few of the accessibility features built-in the present generation mobile devices: VoiceOver and TalkBack, zoom and speaking screens, font adjustments, close captions, etc. This means two things: first, people with disabilities can now choose from a selection of off-the-shelf alternatives for their mobile needs, without paying more. And designers can reach all users with a single interface, not one "main" app or site and one that's "text only" or "downlevel." Both of these are dramatic changes for the better in the field of inclusive design.

These embedded features can be adjusted to each user's preference, and will apply to all native apps on the device. But here is the glitch: each new nonnative application downloaded on the device will not necessarily have accessibility features, and will not always recognize the specific features in place. It is up to each user to adapt the new app, which can be a tedious exercise.

PhoneGap and PhoneGap Build: Take the pain out of developing mobile apps

PhoneGap is a mobile application development framework, based upon the open source Apache Cordova project. It allows the developer to write an app once with HTML, CSS and JavaScript, and then deploy it to a wide range of mobile devices without losing the features of a native app. PhoneGap makes native mobile apps using HTML, CSS and JavaScript for supported platforms: iOS, Android, BlackBerry, Windows Phone 8, Windows 8 and new platforms: Tizen, Qt, Firefox OS, Ubuntu Mobile, Win32. PhoneGap also includes plugins for device native features (alarms, alerts, LEDs, etc.). PhoneGap Build is a cloud-based service built on top of the PhoneGap framework. It allows you to easily build those same mobile apps in the cloud.

The process for building an app in PhoneGap is virtually identical to building a mobile website, except for how it is packaged and deployed. One way of looking at an app written in PhoneGap is as a set of web pages wrapped in executable code. It will function like typical HTML code would on any given host device, including built-in accessibility support—with one exception: for privacy reasons, web content isn't allowed access to assistive technology status. The application programming interfaces (APIs) simply don't exist. This is necessary on the web to keep sites from actively excluding people for their use of accessible technology.

A mobile device, however, is a much more personalized sphere of control than the entire web. The user depends on a higher degree of customizability for their device and the apps they need. So mobile operating systems have their own accessibility APIs available to developers to report whether a specific accessibility feature is engaged, so it can communicate what's happening on the device when it's running, and stay idle when it's not. (Battery life also explains the need for this status.).

Erik Zetterström, ICTenablers, for KenArt Media, Sweden

"I would say that using PhoneGap/Cordova cuts our development costs by 50%, at least. I'm located in Sweden and do mainly development of assistive technology apps.
The project I'm currently working on will be public knowledge on the 12th of November. So I talked with my customer last week and they gave me an ok to spill the beans to you.

My client is KenArt Media they currently have a software suite called Kan Serien/Yes I Can. It is a tool to teach children who are deaf or hard of hearing sign language, lip reading, spelling, reading and hearing. The current software is Windows based and they wish to deploy it on Android and iOS devices, that's when I got involved. I think it is a simple yet great product!"

How it works:

1. The author builds a site with HTML, CSS, JS, images, and adds plugins for non-web functionality
2. Calendar, barcodes, battery, Bluetooth, flashlight - Plugins can also be written separately
3. PhoneGap bundles the site with the operating system's WebView PhoneGap sends the app to the compiler for each operating system
4. The compiler generates a native executable - • .ipa for iOS, .apk for Android, etc,.
5. The author deploys the app through the OS app store or by sideloading

PhoneGap supports most of the accessibility features of mobile browsers. PhoneGap apps run in the operating system's webview which are quite powerful on major mobile platforms. Therefore, users with disabilities will have an experience comparable to using a web based application within the easier confines of a native app. With basic HTML accessibility, any work that the developer does to make the HTML site accessible will carry over to the app as well. For example, if the web page is marked up using headings, VoiceOver users on iOS can set the rotor to headings to navigate through the content. But general web accessibility rules still apply as defined by WAI/WCAG 2.0.

But now that each operating system has its own accessibility support, and the feature sets roughly overlap, cross-platform development tools like PhoneGap were at a functional disadvantage to native toolkits in that they couldn't query or respond to changes in the accessibility status of a given device.

All native apps running on mobile platforms have access to accessibility APIs. They can determine if a screen reader is running or if captions are enabled and so on. PhoneGap apps, because they run inside the web view, cannot get to this info. Therefore PhoneGap apps do not have the capability to enhance the user experience for accessibility users.

That's the reason for the PhoneGap Mobile Accessibility plugin that Adobe Systems has posted on Github:
https://github.com/phonegap/phonegap-mobile-accessibility
The plugin is only to add the capability to these apps that the native apps are supposed to already have. Here are some of the methods available in the accessibility plugin:
1 - On mobile, your own device is (mostly) your private agent, and you can set default font sizes, screen reader & captioning settings, etc. All native mobile apps have access to this data
2 - Mobile apps can adapt more easily to AT status
The PhoneGap Mobile Accessibility plugin adds a new object, MobileAccessibility, to be queried by app developers. This object contains answers to several questions that can't be asked with traditional web APIs:

- Is a screen reader currently running?
- Is closed captioning enabled at an operating system level?
- What level of text zoom has the user selected?
- Has the user chosen inverted colors for higher contrast?
- Has the user chosen mono audio instead of stereo?
- Does the user prefer exploring the screen by touch, rather than navigating sequentially?
- Has the user limited certain buttons or areas of the screen from being activated?

Each of these indicators can provide valuable information to a running app, as they can alter how they look or function. The plugin allows authors to note that one or more of these features is engaged and offer a complementary interface to each user at runtime. For example, an app with audio or video content can ask to see if the user prefers captioning, and if so, it can display captions automatically to go with that content.

In iOS, there is only one screen reader available: VoiceOver. But on Android, and potentially other platforms, users may have a set of tools to choose from. In particular, Android versions 4.0 (Honeycomb) and higher use ChromeVox, a screen reader built using Web technologies, instead of TalkBack to provide enhanced accessibility to the web view. The Mobile Accessibility plugin allows authors to see if either or both are presently in use, and could be extended to support any assistive technology that can be detected. If a supported platform has a screen reader, the Mobile Accessibility plugin can trigger it to speak or stop speaking directly.

Likewise, operating systems that notify users when the page scrolls or its layout changes, the plugin can send a textual notification to the user. The plugin works through a library written in native code for each supported operating system (currently Android, Kindle Fire OS and iOS, with Firefox OS 2.0 and Windows Phone 8.1 under evaluation). This library can query the existing values at the operating system level in order to make them available to the content of the web view. The source code for each operating system is open source, as is the rest of the plugin, and PhoneGap itself. Developers who want to expand to other platforms can write and contribute libraries of their own.

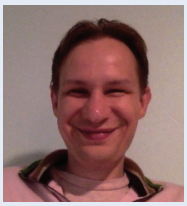
MobileAccessibility

The MobileAccessibility object, exposed by window.MobileAccessibility, provides methods for determining the status of accessibility features active on the user's device, methods changing the text zoom of the Cordova web view and for using the user's preferred text zoom as set in the operating system settings, and methods for sending a string to be spoken by the screen reader or to stop the screen reader from speaking.

Methods

- MobileAccessibility.isScreenReaderRunning
- MobileAccessibility.isVoiceOverRunning
- MobileAccessibility.isTalkBackRunning
- MobileAccessibility.isChromeVoxActive
- MobileAccessibility.isClosedCaptioningEnabled
- MobileAccessibility.isGuidedAccessEnabled
- MobileAccessibility.isInvertColorsEnabled
- MobileAccessibility.isMonoAudioEnabled
- MobileAccessibility.isTouchExplorationEnabled
- MobileAccessibility.getTextZoom
- MobileAccessibility.setTextZoom
- MobileAccessibility.updateTextZoom
- MobileAccessibility.usePreferredTextZoom
- MobileAccessibility.postNotification
- MobileAccessibility.speak
- MobileAccessibility.stop

Victor Tsaran, Technical Program Manager, Google



PhoneGap/Cordova is a wonderful, fast, open source tool to develop apps no matter what the platform is. Adding the mobile accessibility plugin ensures that all apps can detect the specific settings of a particular user, and immediately adjust to them, a great benefit to users for a better experience. The cooking recipes app that my wife and I developed works on all platforms, and identifies ScreenReader, VoiceOver, TalkBack and other accessibility features when the user has activated them in the settings.

Global Initiative for Inclusive Information and Communication Technologies

www.g3ict.org