HTML5Apps

Deliverable D2.2
Future Standards Report 1

30 September 2014

Project	
Grant Agreement number	611327
Project acronym:	HTML5Apps
Project title:	HTML5 for Apps: Closing the Gaps
Funding Scheme:	Coordination & Support Action
Date of latest version of Annex I against which the assessment will be made:	May 24, 2013
Document	
Deliverable number:	D2.2
Deliverable title	Future Standards Report 1
Contractual Date of Delivery:	M12
Actual Date of Delivery:	M12
Editor (s):	Dominique Hazaël-Massieux
Author (s):	Stéphane Boyera, Dominique Hazaël-Massieux
Reviewer (s):	Philipp Hoschka, Marie-Claire Forgue
Participant(s):	
Work package no.:	2
Work package title:	Future Standards
Work package leader:	Dominique Hazaël-Massieux
Distribution:	PU
Version/Revision:	
Draft/Final:	Final
Total number of pages (including cover):	43
Keywords:	Web standards, Web and payments, Web and mobile

DISCLAIMER

HTML5Apps is a project funded in part by the European Union.

TABLE OF CONTENTS

1. INTRODUCTION

The market for mobile applications ("apps") is currently dominated by proprietary platforms (esp. Apple iOS, Google Android). Development targeting these platforms requires tools and programming languages that are specific to each of them. That specificity increases the cost of developing applications that work universally on mobile devices for software developers, which in turns means that consumers may not always find the app they want on the platform of their choice.

HTML5-based "apps" developed using a combination of HTML, CSS and Javascript can solve many of the issues of native apps. This is because HTML5 is platform and device agnostic and thus supported on nearly all platforms, making it easy to move apps from one platform to another and also from one device to another (e.g. mobile to tablet to connected TV to connected car).

HTML5 apps have thus great appeal to developers. However, today, HTML5 cannot be used to fully replace native apps. This is because it is lacking a certain number of functionalities such as rich APIs to interact with devices (e.g. to save files to the local filesystem) or support for handling of payments.

The goal of the HTML5Apps project is to close the gap between native and HTML5 apps through the standardization of missing HTML5 functionality.

Work package 2 of the HTML5Apps project aims at launching new standardization efforts to close Gaps between native apps and HTML5 apps, by looking ahead to plan and execute further standardization activities required to make HTML5 apps competitive with native apps.

Whereas standardization work happens in W3C Working Groups (such as the System Application Working Group that the project is staffing as part of its Work Package 1), the work to prepare and accompany future standardization is often informed via public events known as W3C Workshops as well as in W3C Interest Groups and Community Groups:

- W3C Workshops are events organized by W3C that any interested party can participate in and bring input to; they often mark the first milestone of new work starting its way to W3C standardization.
- W3C Interest Groups are groups formally chartered by W3C to bring input (such as use cases and requirements) to existing groups and to identify the need and scope for potential new groups.
- Community Groups are groups that anyone can create, join and participate in; they allow the wider Web community to collaborate on experimental work, with a simplified path towards formal standardization in W3C should that experimental work prove successful.

The first year of the project in the "Future Standards" WP had two main goals:

- Establishing the foundation work to fill one of the most regularly pointed gaps of HTML5 applications: the ability to use integrated payments.

- Supporting and animating the W3C Web and Mobile Interest Group; this group has for mission to “accelerate the development of Web technology so that it becomes a compelling platform for mobile applications”[1], and thus is the natural home in W3C to gather and define the needs for new Web technologies relevant to mobile.

This report summarizes the work accomplished by the HTML5Apps project toward these goals from October 2013 to September 2014.

[1] Web and Mobile Interest Group Charter: http://www.w3.org/2013/07/webmobile-ig-charter.html

2. Future standards on Web Payments

For the first year of the project, the primary objective was to explore the opportunities to launch standardization activities around one of the topics that have been regularly pointed as gap for HTML5 apps: the ability to use integrated payments. For instance, the Vision Mobile Developer Economics Survey of Q3 2013[2] reported: “web technology lacked the ingredients that would turn a technology solution into a platform: […]a means to distribute, monetise and discover web apps”.

2.1. Identification of Web Payments standardization opportunities

In order to identify the opportunities for W3C to launch standardization activities around Web Payments, the HTML5Apps project organized, as a first step, a workshop on the topic. As reported in details in Deliverable 2.1[3], the event, entitled “Web Payments: how do you want to pay?”, took place end of March 2014 in Paris (see Illustration 1).

Illustration 1 - Banner for the W3C Web Payment workshop

The HTML5Apps staff was the main organizer of this event, and carried out the following tasks:

- Creating a Call for Participation to the workshop,
- Encouraging stakeholders to submit position papers,
- Organizing the review process,
- Running the workshop,
- Creating the workshop report (Deliverable 2.1).

The workshop’s objective was to highlight specific standardization needs in the space of payments, and the need for a place for the payment industry stakeholders and the Web players to meet and agree on the best way to tackle the current challenges in the domain. In that regard, W3C was acknowledged by the participants as being the most relevant place.

During the event, a consensual vision emerged from the discussion. New standards should enable:

- Users to choose their preferred payment solutions across all their devices,

[2] http://www.visionmobile.com/product/developer-economics-q3-2013-state-of-the-developer-nation/
[3] http://html5appsproject.files.wordpress.com/2014/07/d2-1-standardization-workshop-report.pdf

- Merchants to transparently support a growing number of payment solutions,
- Developers to monetize applications more easily,
- New payment solution providers to enter the market more easily with innovative solutions and payment schemes, and
- Society to decrease online fraud.

From a technical perspective, three areas of work will help bridge the apps and payments ecosystems:

1. **Payment transaction messaging**: in order to increase interoperability, it will be essential to standardize the protocols and messages between Web applications and payment providers. This includes information about the transaction (such as merchant ID, amount, and currency) and the digital receipt (proof of payments) for both merchants and customers.
2. **Wallet and Wallet API**: A “wallet” is an application that facilitates the installation and selection of specific payment solutions. Standardizing the interface between Web applications and wallets will allow Web application developers to support any current and future payments solutions transparently.
3. **Identity, Authentication and Security**: To decrease fraud, we will look for building blocks for verifiable identity and secure authentication of the different parties on a payment transaction.

2.2. Web Payment Steering Group charter

While it is too early at this point in time to identify the exact technical elements that need to be developed to implement this above-described vision, the participants of the workshop agreed that it is essential to establish as soon as possible a “**steering group**” to formulate a strategy and roadmap of Web Payments, including existing work at W3C and potential new work.

Based on this output, the HTML5Apps staff organized the creation of a possible future W3C Interest group dedicated to Web Payments to take on the role of a web payment steering group. This consisted in conducting investigations and discussions with various players from the payments and Web industry to evaluate their interest in joining forces within a W3C group and in identifying promising areas to start focusing on.

In order to drive this effort and to start a community building process, the HTML5Apps staff set up and launched a W3C Community Group[4] to develop the charter of the future Web Payment Interest Group. W3C Community Groups are pre-standardization groups that anyone can create, join and participate to.

The charter development process[5] was completed in July 2014; the resulting charter was approved by the W3C Management in August and is currently under review by the W3C members for approval. If approved, the new group is expected to start its activities in October 2014.

The charter identifies 6 specific deliverables detailed below.

[4] http://www.w3.org/community/webpaymentsigcharter/

[5] http://www.w3.org/2014/04/payments/webpayments_charter.html

2.2.1. Web Payments roadmap

The future Web Payments Steering Group is expected to develop a roadmap on Web payments with the following objectives:

- **Identify and review existing, relevant technical standards for payment systems** in terms of e.g. risk management and governance.
- **Identify existing and possibly future issues and challenges of Web payments**, from technical and business perspectives. This includes the identification of the different actors in the payments chain, their position, their business models, their responsibilities, their incentives, etc.
- **Identify a set of scenarios that are in the scope of Web Payments work**, including payments in brick and mortar stores with mobile devices, off-line payments, micro-payments, integration of issues such as “floor-limits” and “stand-in” for specific transaction scenarios should be considered, etc. These scenarios should highlight the interfaces between payment systems and applications as well as the complete transaction flow. They should also highlight interactions with essential external services such as identity providers. It may be appropriate to design a typology of uses cases where a set of cases illustrate in different ways the same element. Such a typology will help separating the overall space in smaller units that could be handled separately.
- **Identify where standards are needed to ease the transparent interaction and integration of existing and future payment methods, and Web applications**. This includes investigating how to:
    - Enable a level-playing field for payers, payees and payment service providers, opening the market for more innovation and competition.
    - Reduce the burden on payers and payees to support multiple payment providers and their selections for a given transaction, along with improved security and customer confidence.
    - Provide more flexibility for payers and payees to use multiple payment instruments.
    - Increase user protection (privacy, fraud, etc.) when paying on the Web as well as reduce payee exposure to risk from fraud
    - Provide more transparency of choice to the user to understand the roles of involved parties, assess the effects of possible fees, and understand the data flow and its implications (e.g. for privacy, governance, etc.)
- **Identify where standards are needed to ease the management and interoperability of bill/utility payments**
- **Identity other services that are related to payments** such as invoices storage, digital receipts storage, warranty, recurring payments, etc.

2.2.2. Web Payments terminology

The second objective of the group is to develop and adopt a dedicated terminology related to Web payments. This terminology will be based on existing terminology that has been established by a variety of international organizations and standards. This includes e.g. UNCITRAL terminology, World Bank Terminology, ISO20022 or ISO29115. It will also potentially extend and refine them to cover needs identified in new use-cases or scenarios

2.2.3. Wallet and Wallet API

The topic of Wallet is one of the core elements of the future group. The work in this area will consist of identifying the role and the place of a digital wallet in the payment process in the different scenarios identified in the roadmap (e.g. online and onsite payments). This includes the investigation of Wallet at the customer end as well as at the merchant end (connected to merchant's checkout/payment option). The objective is to define an open framework that encourages innovation in digital wallets and leverage interoperability with merchant sites. The expected output includes the identification of the functionalities of wallets and the interactions with the different stakeholders, the identification of needs for standards, and the identification of requirements to enable integration of new payments schemes and ancillary services, such as loyalty cards or coupons.

2.2.4. Payment Transaction Messaging

The Web Payments Steering Group aims to identify and review existing, relevant technical standards related to transaction messaging, identify requirements and constraints to define a standard way for merchants to describe transaction contents and merchant identification (aka “tokens”), identify requirements and constraints to define a standard way for payment service providers to communicate transaction results back to the merchants and users and identify requirements and constraints to define a standard way to initiate payment process within a web application. This includes the possible provision of customer information (shopping attributes) such as geolocation, time of purchase, or any other information that might be requested by the payment providers to e.g. detect fraud.

Finally, the group will work on identifying requirements and constraints to define a standard way for payment service providers to communicate specific account information such as account balance, transaction history, etc.

2.2.5. Identity, Authentication, and Security

The Web Payments Steering Group will identify and review existing, relevant technical standards for authentication, secure transactions and identity provision, identify potential improvement to Web user-agents (a Web browser, a hybrid app, or an installed Web application) to enable improved authentication using various technologies from multi-factor authentication to secure-elements, to smartcard-based authentication.

It will also review existing Identification mechanism and identity providers on the Web and evaluate whether they fit with payments requirements in terms of privacy and security. The group will develop requirements and use-cases otherwise to seed new work in the area. A particular attention will be put on privacy aspects, and information exchange between identity providers and payment system providers

The group will work to towards identifying user data protection and user privacy issues as well as the management of data provisioning required by regulation and by anti-fraud detection processes.

Finally, the group will explore the provision and access to basic user and payment provider information via the Web in a way that is easy to synchronize across devices and easy to share with various merchants given authorization by the customer.

2.2.6. Reviews, Comments and Provide requirements

Finally, the Web Payments Steering Group will be conducting review, comment and will provide requirements to standards and other related documents developed by W3C and external groups related to Web Payments.

2.3. Upcoming standardization work

2.3.1. Web Wallet APIs, Access to Secure Element

While the scope and the mission of the steering group is wide, the various stakeholders that have expressed interest to join (see below) are primarily interested in identifying and prioritizing work items, and launch as soon as possible technical working group(s) that may have a significant impact on the domain in the short term. At this point in time, the work on Wallet seems to be one of the first candidates for such development. The vision behind Wallet is summarized in Figure 1.

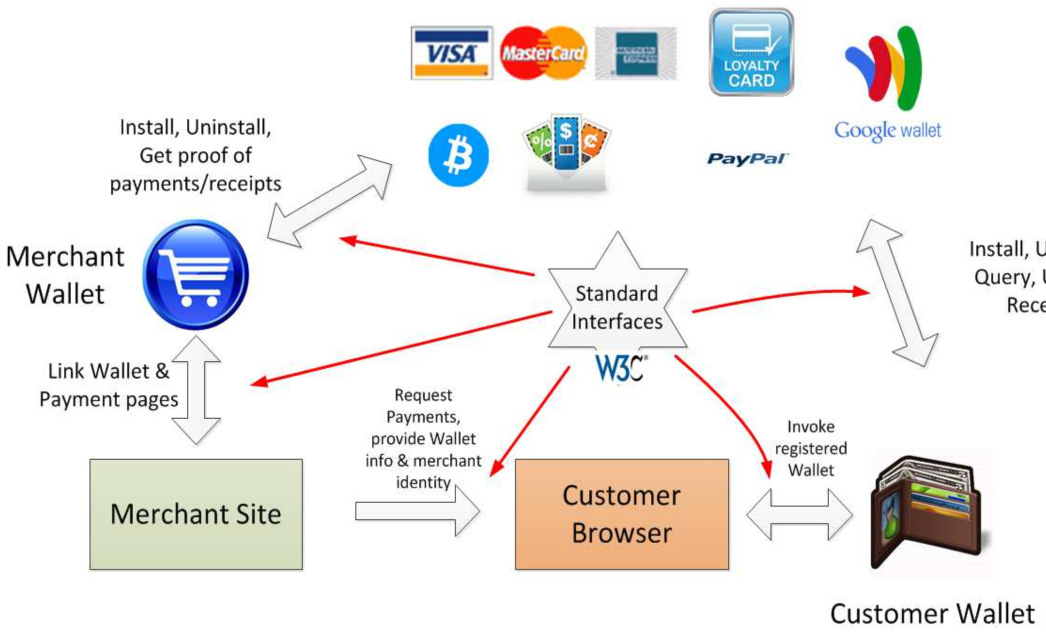

VISA
MasterCard
LOYALTY CARD
Google wallet
PayPal
Install, Uninstall, Get proof of payments/receipts
Merchant Wallet
Standard Interfaces
W3C
Install, Uninstall, Query, Use, get Receipts
Link Wallet & Payment pages
Request Payments, provide Wallet info & merchant identity
Invoke registered Wallet
Merchant Site
Customer Browser
Customer Wallet

Figure 1 - Standardization of Web APIs to wallets

In such model, standardized interface between Web applications and wallets will enable innovation in the space and a greater adoption by merchants and customers. In particular, we believe that, by providing a unified interface for all payment transactions, the user wallet will increase trust from the user perspective and therefore lower the rate of cart abandonment[6] (close to 97% on mobile devices today[7].)

Moreover, by preventing the exchange of sensitive information between the customer and the merchant (e.g. credit card information), wallets could be a way to change the model

[6] Cart abandonment : when a user fills in an e-commerce site shopping basket and does not finalize the transaction and does not buy the content of the basket.
[7] http://seewhy.com/97-shopping-cart-abandonment-rate-mobile-devices-concern-you/

of credit card payment on the Web, and reduce the fraud such as e.g. the one illustrated by the story of security breach at the US retailer Target[8] and similar security breaches.

The exact details of the technical elements that need to be standardized to enable this vision will be part of the discussions of the Web Payment Steering Group (W3C Interest Group in W3C terminology).

Other major areas of interest include topics related to authentication and access to hardware cryptographic elements from Web applications. The SysApps WG has currently this item in its charter (see deliverable D1.1) and an unofficial WG draft[9] is available. This is of high interest to major EU players. For instance, Gemalto (a world leader in smartcard technology) is actively contributing to the standardization of this technology.

The Web Payment Steering Committee will work with the group driving this work item to ensure that the requirements related to Web Payments can be addressed by the technologies and standards they are developing.

2.3.2. EU Commission Participation (DG Competition)

The EU commission (Alexander Gee, Deputy Head of the Payments Unit for the DG Competition) delivered a keynote speech at the workshop[10].

In his talk, Mr Gee promoted the development of initiatives working toward making payment on the Web work better, based on open non-discriminatory standards.

He described in details the approach of the EU commission focusing on interchange fees that are considered as one of the biggest barriers to competition in payments market. Today the fees are around 1%, and the EU objectives is to bring them down to 0.2 to 0.3%. The major issue for interchange fees is that they are hidden to the customer.

Mr Gee highlighted the fact that the current e-payment market is not competitive and not open to non-bank players, and the EU is striving for the ecosystem to change. The core focus on the EU is to ensure that the competition is open and based on transparency, particularly on fees. This would lead to more opportunities for non-bank parties to compete in the market. In that regards, The European Parliament’s plenary session on 2 - 3 April 2014 voted the proposed Directive on payment services in the internal market (PSD2) and the proposed Regulation on multilateral interchange fees for card-based payment transactions (MIF Regulation). These directives fixe the interchange fees and describe the requirements for third-party payment service providers (TPPs).

The work plan developed in the Web Payments charter follows exactly the same objectives of creating an open interoperable framework for Web payments enabling competition and innovation from new (in particular non-bank) players. It is important to note that the key requirements for TPPs are at the core of the work of the W3C Web Payments Activity. This covers in particular:

- Limited access to customers account information (receipt of payment order and adequate funds): The W3C Web Payments charter includes the definition of the message exchanges between TPPs and web applications and will ensure that the regulatory constratins are implemented in the proposed technology standards.

[8] http://www.zdnet.com/target-confirms-breach-40-million-accounts-affected-7000024499/
[9] http://opoto.github.io/secure-element/
[10] http://www.w3.org/2013/10/payments/slides/gee.pdf

- No storage of confidential data or credentials: One of the core work item of the Web Payments activity is on tokenization of transaction, that prevents the communication of sensitive customer informations (such as credit card information) to merchants and TPPs. The standardization of these tokens and their interoperability across TPPs will enable the implementation of this regulatory constraint.
- Strong authentication: The W3C Web Payments activity has a specific work item on strong authentication based on hardware tokens, secure elements or biometric elements. The development of technology standards on how to use these hardware elements to provide a strong authentication for financial transactions is one of the core items of the group, and will fulfill the requirements adopted by the EU parliament

Mr Gee reviewed and commented on the draft charter[11] and is interested to join this work.

2.3.3. European Industry Involvement

In terms of participants, a big part of the projects effort since the workshop in March has been dedicated to identify and to bring at the table relevant stakeholders for this future new activity.

At the time of this report, major European Banks have expressed interest to join the W3C payments work. This includes

- BPCE (France)
- BNP-Paribas (France)
- ING (Netherland)
- Rabobank (Netherland)
- Abn Amro (Netherland)
- BBVA (Spain)
- HSBC (UK).

Other actors of the payment industry have also already joined W3C for this work. This includes Ingenico, Lyra Networks or Canton Consulting. We are currently engaging discussions with major e-commerce companies such as Zalando.

2.3.4. International Involvement

Apart from the European players, we are also reaching out to other international players to ensure that the work that will be developed can be applicable and usable in all regulations across the globe. The fact that the US Federal Reserve has joined W3C to participate in this work, and that the World Bank participated in the workshop and has expressed interests to join are promising signs for the wide applicability of this work.

All the elements have now been put in place to start the work and enable a new series of standards in the area of Web Payments, making the Web a better platform for monetizing content and applications.

[11] http://lists.w3.org/Archives/Public/public-webpaymentsigcharter/2014May/0025.html

3. Paving the way towards new standards

3.1. W3C Web and Mobile Interest Group

The HTML5Apps project is staffing the W3C Web and Mobile Interest Group. As defined in its charter, this group's mission is "to accelerate the development of Web technology so that it becomes a compelling platform for mobile applications."

This mission has been aligned with the goals of the HTML5Apps project in general, and of the "Future Standards" work package specifically: one of the primary role of this group is to evaluate and refine the need for new standards to make Web technologies competitive with native developments, and increase their interoperability.

By investing resources in this Interest Group, the HTML5Apps project leverages W3C's process and membership to increase the impact of its work in the identification and definition of the future standards needed to make HTML5 apps competitive with native apps.

Co-chaired by a representative of GSMA (the mobile operators association) and a representative of Mozilla (maker of the Firefox browser) the Web and Mobile Interest Group started the gist of its work at the same time as the HTML5Apps project itself started. The project is funding the effort of a W3C team contact in this group, Dominique Hazäel-Massieux, also WP2 leader.

During the first year of the project, the tasks conducted by the HTML5Apps staff in the Web and Mobile Interest Group can be classified under four categories:

- Getting the group starting and running,
- Laying the ground framework for identifying and defining new standard opportunities,
- Driving and contributing to the identification and definition of new standard opportunities,
- Identifying and documenting barriers to the adoption or the definition of new standards for HMTL5 applications.

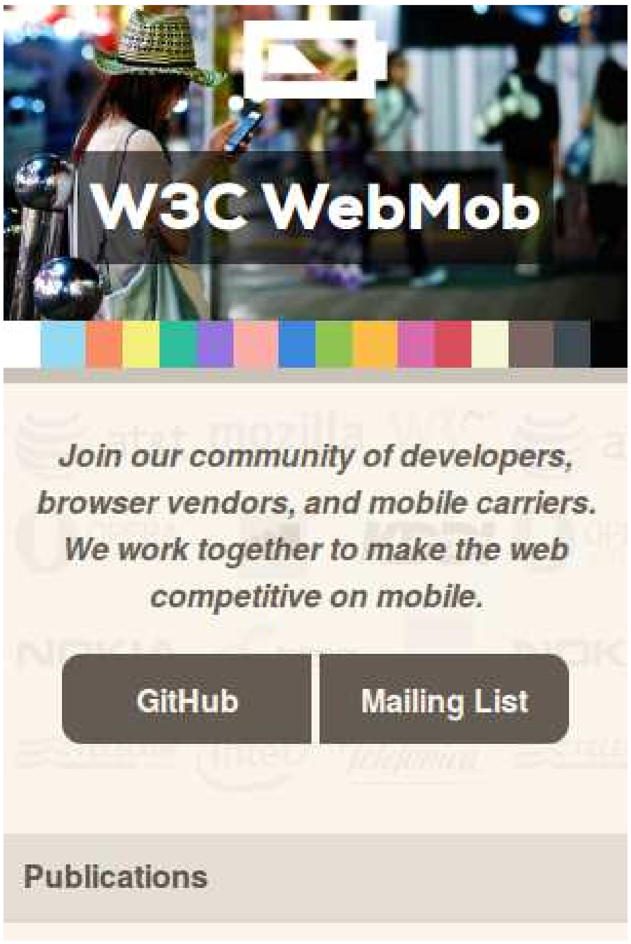
As a the Web and Mobile Interest Group was just starting, the HTML5apps staff HTML5Apps staff worked with the Interest Group co-chairs to get it up and running. This involved in particular:

- Setting up the group home page[12] (see also Illustration 2) and associated tools (public mailing list[13], wiki[14], issue tracking system, Github account[15], Twitter account). In particular, to make it easier for group participants to contribute, the group's home page has been set up as a Github repository of its own[16];
- Organizing a schedule for monthly teleconferences (ten of which have been held so far);

---

[12] http://www.w3.org/Mobile/IG/
[13] http://lists.w3.org/Archives/Public/public-web-mobile/
[14] https://www.w3.org/wiki/Mobile/
[15] https://github.com/w3c-webmob/
[16] https://github.com/w3c-webmob/w3c-webmob-website

- Inviting some of the critical players to participate the group (browser vendors, operators, mobile apps developers, device makers);
- Setting up the group first face-to-face meeting as part of the W3C annual Technical Plenary week, organized in November 2013 in Shenzhen, China.



Join our community of developers, browser vendors, and mobile carriers. We work together to make the web competitive on mobile.

Illustration 2 - Web and Mobile Interest Group home page as viewed on a mobile browser

The Web & Mobile Interest Group primarily operates via its public mailing list, public-web-mobile@w3.org (publicly archived on the Web[17].)

As alluded above, the group had a face-to-face meeting as part of the W3C Technical Plenary week in Shenzhen China in November 2013, and has had since 10 monthly teleconferences. These meetings, whose agendas the HTML5Apps project helps establishing, allow the group to evaluate its progress, identify new topics of work, and share news and ideas relevant to its mission.

[17] http://lists.w3.org/Archives/Public/public-web-mobile/

3.2. Identifying Future standards

3.2.1. Future standards framework

Numerous articles, reports and opinions have been written to compare the relative advantages and drawbacks of HTML5 applications to their native counterparts. To ensure the Web and Mobile Interest Group would be in a position to address the most relevant among the drawbacks, the HTML5Apps project invested resources in laying out the ground frameworks needed to categorize and analyze these differences.

As a first step in that direction, the HTML5Apps project contributed and presented a framework document to the group: "Gap Analysis between Web and Native on Mobile."[18]

This framework organizes the gaps that had been identified in previous research in two main categories:

- gaps in user experience,
- gaps from a content or service provider perspective.

The former are gaps that affect the way an end-user would interact with a given application – user experience is a frequent theme in comparisons between native and HTML5 apps. The latter are gaps that would prevent or encumber a content or service provider from using HTML5 technologies to develop and deploy their application.

The document, divided in 6 parts, looks at each of the aspects of user experience (resp. provider requirements) that have been known to be particularly important on mobile devices, compares how Web and native fare in these aspects, and sketches some possible actions to reduce the gaps between them.

Contributed at the early stages of the Interest Group, and presented both during one of the group's teleconferences and at its face-to-face meeting, this framework remains one of the cornerstone of how the group approaches its research on HTML5 application gaps.

In addition, to ensure the group would properly keep track of new research and newly identified gaps, the HTML5Apps project heavily contributed to the content and the organization of the research tracker[19] of the Web and Mobile Interest Group, maintained as part of its wiki. This collection of articles, surveys and analysis has already started to get recognition in the wider community as a reference point.

3.2.2. Laying the ground for future standards

Out of the follow up discussions that emerged from the gap analysis report, the Web and Mobile Interest Group identified a set of topics that needed to be worked on, and created task forces[20] that would dedicate efforts towards that work.

Among these task forces, the HTML5Apps project contributed efforts toward the topics described below.

[18] http://w3c-webmob.github.io/gap-analysis/
[19] https://www.w3.org/wiki/Mobile/articles
[20] https://www.w3.org/wiki/Mobile/Work

Network Information API

The Network Information API had been under development by the W3C Device APIs Working Group for several years, but had failed to progress due to lack of clarity on the exact use cases that it needed to fulfill.

The Web and Mobile Interest Group thus worked on defining these use cases, deriving requirements from them. The HTML5Apps project contributed both to use cases, documenting what native applications use and do in this space, as well as to the requirements definition. The group published its results as W3C Interest Group Note in April 2014[21].

Following that analysis, browser vendors have now started to build and deploy experimental implementations of the API to gather explicit feedback from developers on how the said API helps fulfill these use cases.

Screen-lock API

Prompted by a request from an application developer, the HTML5Apps project looked into[22] the needs and requirements for an HTML5 API to prevent the default automatic screen lock on mobile devices.

In order to save battery, most mobile devices turn their screen off and lock the device when the user is not directly interacting with the device. However, this default behaviour is not appropriate in certain specific use cases: For example, when users are watching a longer video, they are unlikely to interact with the device (e.g. touch the screen). Obviously, in this use case, turning of the screen due to user inactivity is not appropriate, and needs to be prevented via an API controlling screen-lock.

The HTML5Apps project analysis has shown that several Web Operating Systems already provide APIs for controlling screen-lock are already deployed in significant way in non-standardized HTML5 ecosystems, and this API is thus a relevant target for future standardization.

After gauging increased interest from developers and browser vendors around that feature[23], the HTML5Apps project proposed a specific technical approach to offer this as an API on the Web[24], including a well-defined permission model, and contributed to the development of a set of use cases and requirements against which possible solutions ought to be judged. These use cases and requirements were published as a W3C Interest Group Note in August 2014[25] (see also Appendix 2: Wake Lock: Use cases).

Following this proposal and proposal from others Web and Mobile Interest Group participants, this work item has been added to the Device APIs Working Group, and is expected to be published as a W3C First Public Working Draft in the upcoming weeks.

Identity management and authentication

Given the personal nature of mobile devices, and given the increasingly customized experience that mobile applications strive to provide, identity management and

---

[21] http://www.w3.org/TR/netinfo-usecases/
[22] http://lists.w3.org/Archives/Public/public-web-mobile/2014Feb/0007.html
[23] http://discourse.specifiction.org/t/allow-developers-to-control-wake-lock-aka-disable-auto-dimming/72
[24] https://github.com/w3c/wake-lock/commit/3f1fc9033c855ec0fd4338fbf4bb21564fc2bc9a
[25] http://www.w3.org/TR/wake-lock-use-cases/

authentication are critical aspects for the development and deployment of competitive HTML5 applications.

As a result, the HTML5Apps project proposed the creation of a dedicated task force to look at topics around identity management, and recruited participants that would be in a position to contribute to this discussion. While this work is just starting, it is likely to highlight important needs for future standards.

Management of Audio Devices

Mobile devices are often used as portable media players, and as such, can be used in different audio configurations: by means of the small range audio ”earpiece” used when making calls, via the more powerful embedded loudspeaker, with a headset via the jack plug or via Bluetooth, connected to an external loudspeaker.

Audio-oriented mobile applications need to detect and react to these configuration changes: for instance, some apps will want to mute their audio when unplugging a headset, where others will want to use the loudspeaker.

The HTML5Apps project reviewed the existing capabilities of the Web platform in that regard, and laid out a plan[26] for identifying gaps in these capabilities. Further work will be needed to determine how and when these additional capabilities can be specified.

Bluetooth LE integration in browsers

Most mobile devices come with the ability to connect with other devices via Bluetooth, a short-range radio communication standard. The devices that can be connected to via Bluetooth are very diverse, from screens and audio output to heart rate monitors and glucometers.

The recent spread of one of the latest evolution of this standard, called Bluetooth Low Energy (often referred as Bluetooth LE) is expected to significantly raise the number of applications that rely on this protocol to interact with low-power devices and objects. Among others, Apple’s iBeacons, which facilitates in-door location and contextualization of an app, have received a lot of attention from mobile developers.

The HTML5Apps project started to review how the Web platform currently deals with Bluetooth in general, and how the characteristics of Bluetooth LE could be understood in the context of future APIs[27] as deployed in Web browsers.

Subsequently, a new W3C community group was started in July 2014[28], where follow up work on use cases and pre-standardization technical discussions has now started.

3.2.3. Mapping the space of mobile platform features

In addition to gathering input from developers on gaps they have encountered in developing Web application on mobile, the HTML5Apps project started a more comprehensive approach to evaluate what features are available in native ecosystems but missing from the Web platform.

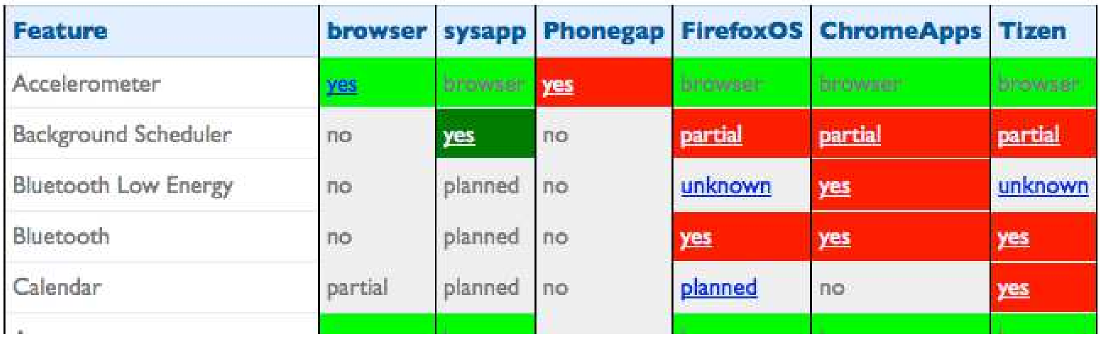
[26] http://lists.w3.org/Archives/Public/public-web-mobile/2014Jul/0013.html
[27] http://www.w3.org/TR/wake-lock-use-cases/
[28] http://www.w3.org/community/web-bluetooth/

To that end, the HTML5Apps project started a repository[29] documenting which features are available on popular native environments (iOS, Android, Windows Phone 9) and on popular HTML5-based Web Operating Systems (Firefox OS, Chrome Apps, Tizen), and for each of these, whether an equivalent is available or under development for browser-based or HTML5-OS standard based environments.

As of September 2014, that repository documents 35 such features, with direct links to the relevant documentation for each platform, and has been well received by the broader community (as illustrated by the 12 "watchers" on github and the 60 stars the repository currently holds).

The data collected in this repository have already been specifically re-used in the work previously alluded to on the screen wake lock API and on the Bluetooth research the project started.

The data also made it possible to generate an overview comparing the availability of certain features already available on existing HTML5-based platforms (an extract of that view is presented in Figure). This allows among other things to identify primary candidates for standardization: features that are widely available in existing HTML5-based platforms but are not under standardization would likely be of interest for platform vendors to collaborate on, since they already have experience with implementing and documenting APIs for them.



Feature	browser	sysapp	Phonegap	FirefoxOS	ChromeApps	Tizen
Accelerometer	yes	browser	yes	browser	browser	browser
Background Scheduler	no	yes	no	partial	partial	partial
Bluetooth Low Energy	no	planned	no	unknown	yes	unknown
Bluetooth	no	planned	no	yes	yes	yes
Calendar	partial	planned	no	planned	no	yes

Figure 2 - Extract of the HTML5-based platforms fragmentation matrix (for a detailed explanation, please refer to "HTML5-based platforms fragmentation"[30])

The HTML5Apps project will continue to document these features and new ones, and to exploit the results of their analysis to determine the path towards new standards for HTML5 applications.

3.3. Reducing barriers between hybrid and Web applications

Several mobile developer surveys[31] indicate that an increasing number of developers use HTML5 technologies to develop so called hybrid applications, i.e. applications that are packaged and distributed like native applications, but whose main components are developed using HTML, CSS and JavaScript.

[29] https://github.com/w3c-webmob/web-api-gap/
[30] http://lists.w3.org/Archives/Public/public-web-mobile/2014May/att-0023/html5-fragmentation.html
[31] Most recently, the Vision Mobile Developer Economics Q1 2014: http://www.visionmobile.com/blog/2014/02/developer-economics-q1-2014/

The advantage of hybrid applications for developers is that they can build and maintain a single code base, and obtain native applications that will work seamlessly on a wide range of operating systems - thanks to the cross-platform nature of HTML5. To achieve this, hybrid application frameworks provide their developers with JavaScript APIs to integrate with the native capabilities of the underlying operating systems, and build the bridges between these APIs and each and every platform they support.

As illustrated in Figure extracted from a taxonomy of mobile application platforms submitted to the Web and Mobile Interest Group (see Appendix 1: Taxonomy of Mobile Applications), these hybrid applications play an important role in ensuring that HTML5 serves as a leading development platform on mobile: they make HTML5 part of the toolkit that developers can make use of throughout the continuum of application development approaches.

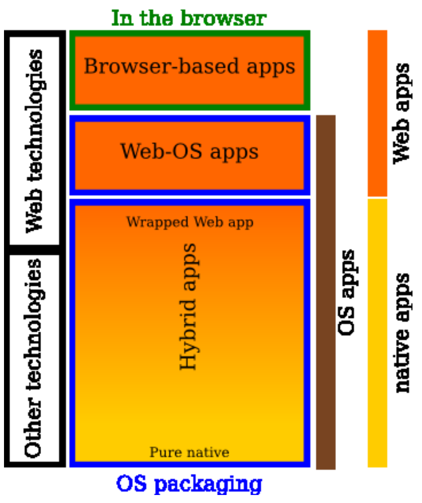


Figure 3 - The continuum of Web and native applications (see Appendix 1: Taxonomy of Mobile Applications)

This means that HTML5 developers can make use of their skills to distribute native applications, and thus be part of that market even when their customers require a native solution. It also allows to distribute HTML5 applications on platforms for which a given feature or API has not been made available in the browser yet.

Finally, these hybrid frameworks provide a compelling platform to get feedback on which new APIs would be most useful, or on how well current Web APIs fulfil the goals of a wide range of applications.

Among the frameworks used to develop these native applications, the most popular one is Cordova, an open-source project on which Adobe PhoneGap’s toolset is built – for historical reasons, Cordova and PhoneGap are often used interchangeably to designate the framework. For instance, up to 6% of applications available on the Google Play Store have been found to have been built with PhoneGap[32] for an overall total of 9.2% hybrid applications, making up nearly 2/3 of hybrid applications.

Given the importance of hybrid applications and the prominent role of Cordova in that space, the HTML5Apps project decided to invest in establishing effective

[32] See *A Measurement Study of Google Play* http://velvetpulse.com/assets/playdrone.pdf

communications with the project to ensure the APIs they provide to developers can be used to inform the work in W3C on APIs for mobile browsers.

Indeed, one of the results of the mobile APIs survey described previously was that among the HTML5-based mobile platforms, Cordova had over time diverged the most from the APIs developed in W3C: among the 9 features for which APIs exist both in W3C and Cordova APIs, only 1 was an exact match[33].

The HTML5Apps project took the following actions:

- We contacted the Cordova project leaders to confirm their interest in aligning the W3C and Cordova APIs;
- Following their recommendations, we started by comparing in details the set of APIs, and logging bugs in the Cordova issue tracking system[34] when we found APIs that ought to be the same but were not;
- We then had a joint call with one of the Cordova committers to establish a strategy toward the resolution of these bugs[35], as well as to determine how the input of the Cordova developers could be properly taken up in the relevant W3C Working Groups;
- In particular, we decided to start the convergence process with a fairly simple and stable API on the W3C side, the Vibration API; doing so would allow us to establish a process for further collaboration without getting slowed down by potential difficult technical issues, or frequent needs to update the code base;
- The HTML5Apps project then collaborated with the Cordova committer on a detailed plan of attack for the Vibration API[36], which was then posted both to the Cordova mailing list and the relevant W3C mailing lists.

This plan of action has already provided some very encouraging results:

- The vibration API now shipped with the latest version of Cordova is the same as the one specified by W3C;
- In the process of implementation the said API, the Cordova developers sent direct technical feedback to the relevant W3C Working Group[37], that allowed to fix an edge case not covered by the specification;
- Several browser vendors (Mozilla, Microsoft, Google) have started participate in the Cordova project to the efforts of convergence[38];
- The Cordova project then moved on to start alignment on the Battery API, during which they again provided useful feedback to the corresponding W3C specification which was updated accordingly[39].

[33] http://lists.w3.org/Archives/Public/public-web-mobile/2014May/att-0023/html5-fragmentation.html

[34] For instance, bug report on the battery API: https://issues.apache.org/jira/browse/CB-6065; all the bug reports are listed in http://lists.w3.org/Archives/Public/public-web-mobile/2014Feb/0051.html

[35] http://lists.w3.org/Archives/Public/public-web-mobile/2014Feb/0051.html

[36] http://lists.w3.org/Archives/Public/public-device-apis/2014Jun/0037.html

[37] http://lists.w3.org/Archives/Public/public-device-apis/2014Jul/0059.html

[38] See for instance Microsoft automated audit of the divergences http://lists.w3.org/Archives/Public/public-device-apis/2014Jun/0040.html

[39] http://lists.w3.org/Archives/Public/public-device-apis/2014Jun/0068.html

3.4. Documenting and reducing barriers to future standards

Beyond identifying needs for future standards, the HTML5Apps project also put efforts in documenting and reducing barriers that make the development or adoption of new standards in the fields of HTML5 applications more difficult.

Indeed, once a need for a new standardized feature has been identified, there can be many obstacles on the way to getting that feature specified and implemented. These obstacles create delays before mobile applications developers can actually start using these features.

The pace of standardization has often been pointed as a disadvantage that HTML5 application developers have to suffer from compared to their native counterparts whose speed of innovation is under fewer constraints, since they are driven by a single vendor.

By documenting and, over-time, reducing some of these barriers, the HTML5Apps project aims at streamlining the path from idea to product.

Over the first year of the project, the issues that we identified as most frequently slowing the progress of new standards have been linked to **security and privacy considerations**.

Indeed, the traditional model in which Web applications operate, inside the browser, assumes very limited trust in the applications that are run, and very careful gating of additional privileges grants. Since a Web browser must be able to display any random Web page the user might have followed a link to, it needs to display and run that Web page without exposing the user to harm, and thus assume a Web page or Web app is fundamentally not trusted.

While that model has proved very rich for the wide deployment of the Web across many devices and operating systems, it makes it very hard to bring features to HTML5 apps that require trust: access to the user personal data, access to sensitive sensors (e.g. camera), ability to run background services, etc. The Web Geolocation API has shown as early as 2008 what can be gained from access to these features, and offered a first approach in managing access to these sensitive features, but that approach hasn't proved yet easy to scale to the many additional features the platform needs.

Most W3C Working Groups developing sensitive APIs are faced with this challenge. The HTML5Apps project started two efforts in the Web and Mobile Interest Group that are aimed at making that challenge easier to tackle:

- A **joint task force between the WebMob IG and the W3C Web Security Interest Group** has been started, with a particular focus on analyzing the overall security life cycle of mobile applications in HTML5; this includes secure data storage, isolation of data stores, management of keys and certifications.
- The HTML5Apps staff conducted a thorough **review of the management of permissions to sensitive APIs across the whole Web platform** in a Github repository[40]. That survey resulted in a summary table[41] showing the various approaches taken by various Working Groups in managing permissions (see also Appendix 3: Matrix of Permissions Usage in the Web Platform). This table was presented to the W3C Technical Architecture Group (TAG) in January 2014

[40] https://github.com/dontcallmedom/web-permissions-req/

[41] http://dontcallmedom.github.io/web-permissions-req/matrix.html

(within W3C, the TAG is responsible for supervising the broad technical evolution of the Web platform). The TAG agreed to the need of stronger coordination and supervision in this space, and to work with the Web and Mobile Interest Group on the topic.

Both of these topics are at heart of some of the debates in the W3C Systems Applications Working Group (staffed by the HTML5Apps project as part of its work package 1 on APIs). In particular, our review of the permissions as used on the Web was an important input to the meeting dedicated to the management of permissions[42] organized by the System Applications Working Group in Paris in September 2014.

The HTML5Apps project expects to continue to put efforts on that work, especially in the context of the follow up actions determined at that meeting, as detailed in the HTML5Apps deliverable D1.2.

[42] http://www.w3.org/2014/07/permissions/

4. Summary and Next steps

During its first year, the HTML5App project laid the ground for many important work items for the future of HTML5 applications:

- The HTML5Apps project organized a **successful workshop on Web payments** and did the follow up work in **chartering a steering group** to pave the path for integrated payments on the Web. This opens up a bright future where HTML5 developers can sell their content and services as easily as native developers have enjoyed.
- The **set up and operation of the W3C Web and Mobile Interest Group** by the HTML5Apps project have created an active forum where the needs for new standards are brought up, discussed and formalized in reference documents (use cases, requirements) that can then serve as basis for the actual development of these standards. Five of these areas of developments have already been identified, three of which have already led to new technical proposals for standardization.
- The HTML5Apps project made several significant contributions to establish a **formal framework to evaluate and identify new standardization needs**, esp. via a comprehensive review of other native and HTML5-based platforms on the market.
- The project also identified the need for and has set up **collaboration with the popular Cordova hybrid application framework** to facilitate the migration between hybrid and Web-based development approaches. The first results of that collaboration (one API aligned, another under alignment, better technical communication between W3C and the Cordova project) are very encouraging.
- Having **determined important challenges facing the development of new standards for HTML5 applications around security and privacy**, the project started a systematic review of the existing approaches in the Web platform to face these challenges, presented it to the main architectural body of W3C, and provided it as input to a critical meeting on the topic organized by the W3C System Applications Working Group (also staffed by the project as part of its Work Package 1).

In the upcoming year, the project will continue and finalize this work as follows:

- The project will support the start and set up of the Web and Payments Steering Group, and provide a W3C Staff Contact as the group begins its work to establish the roadmap for the first technical standards in this space.
- The project will continue staffing the Web and Mobile Interest Group, and ensure progress on the already identified future standards, while looking for opportunities and needs for additional standards. The project aims in particular at providing a per-category of applications (e.g. news, health, productivity apps) view of the gaps, as a way to determine relative priorities of these gaps, as well as to facilitate the collection of input from relevant developers.

- The project will provide continued support to the collaboration with the Cordova project and work to ensure that collaboration gets integrated as the normal mode of operation on both the W3C and Cordova sides.
- The project will invest in the follow up work on permission management that was discussed and identified during the relevant W3C System Applications Working Group meeting, to ensure the technical challenges posed by these questions don’t impede the future progress of the platform.
- The project will also put further efforts in the broader question of managing the secured operations of Web-based applications.

APPENDIX 1: TAXONOMY OF MOBILE APPLICATIONS

Introduction

The many mobile ecosystems available on the market today provide a number of different development approaches for applications.

The fragmentation created by that diversity of programming languages, SDKs, packaging formats has brought to the front the possibility to use Web technologies as a way to address as many platforms as possible.

The mix and match of Web technologies and so-called native ones have led to many comparison between these different development approaches, often refered to as Web applications, hybrid applications and native applications.

But the boundaries between these various approaches are often at best ill-defined. This document proposes a taxonomy of these different approaches to ensure a common vocabulary can be used in making these comparisons.

Taxonomy

We define the following terms to help navigate the many ways in which Web technologies are used to develop applications on mobile:

- **Mobile application**: an application that runs on mobile devices
- **Web application**: an application entirely built with Web technologies
- **Default technology stack**: set of technologies provided by a mobile operating system to build applications
- **OS application**:an application built with the operating system default technology stack
- **Web-based operating system**: an operating system where OS applications can be developed using only Web technologies (even if the default technology stack goes beyond the Web)
- **Native application**: an OS application that uses non-Web technologies
- **Browser-based application**: a Web application that can be run in a browser
- **Web-OS application**: a Web application that runs in a Web-based operating system
- **Hybrid application**: a native application that is developed partly with client-side Web technologies; this ranges from a Web app simply wrapped into a native packaging, to a mostly native app that uses HTML/CSS rendering in a few places

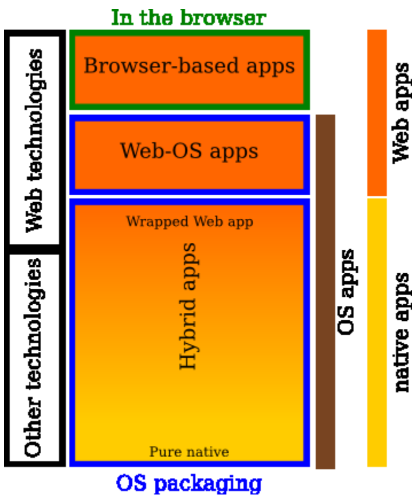


Figure 2: A taxonomy of the various approaches available to develop mobile applications with Web technologies

Appendix 2: Wake Lock: Use cases

W3C Interest Group Note 14 August 2014

This version:

http://www.w3.org/TR/2014/NOTE-wake-lock-use-cases-20140814/

Latest published version:

http://www.w3.org/TR/wake-lock-use-cases/

Latest editor's draft:

https://w3c-webmob.github.io/wake-lock-use-cases/

Editors:

Marcos Caceres, Mozilla

Natasha Rooney, GSMA

Dominique Hazael-Massieux, W3C

Repository:

We are on Github.

File a bug.

Commit history.

Abstract

This document illustrates the use cases a mechanism to control the power-saving state of a device would enable on the Web platform.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This is a work in progress. You can contribute to this specification by contributing additional use cases through our GitHub repository.

This document was published by the Web and Mobile Interest Group as an Interest Group Note. If you wish to make comments regarding this document, please send them to public-web-mobile@w3.org (subscribe, archives). All comments are welcome.

Publication as an Interest Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

The disclosure obligations of the Participants of this group are described in the charter.

This document is governed by the 14 October 2005 W3C Process Document.

Table of Contents

1. Introduction

The web platform currently lacks a means to prevent a device from entering a power-saving state (i.e., some means that prevents an aspect of the system from "going to sleep"). As this document tries to demonstrate, there are strong use cases where applications need to temporarily prevent some aspect of the device from entering a power-saving state. As this functionality is common on other platforms, this document tries to make the case that this functionality should also be made available to the web platform.

To understand the use cases that motivate an application from preventing a device from entering a power-saving state we examined a set of applications running on iOS, Android, and Firefox OS. By looking at these native applications, and the conditions under which they prevent a device from entering a power-saving state, we've come up with a set of requirements. We hope that the web community can address these requirements by creating a specification that affords similar functionality to the web platform.

2. Why we need wake locks on the Web

For tasks such as watching a video full-screen user agents have, for a long time now, prevented the screen from going to sleep automatically. Naturally users would be annoyed if they were watching a movie and constantly needed to move a mouse or poke at the screen of a device to stop the screen from going black.

2.1 Keeping the screen awake

As computing devices have become increasingly mobile, users find themselves relying on mobile devices to assist them with everyday tasks where it is not always practical or desirable to touch the device to keep it "awake". These can be simple things, like using the phone as a flashlight, to complex tasks such as cooking or allowing the phone to navigate us to some destination while driving. In addition, touch enabled devices have become commodified enough that they can be used as interactive signage. Some of these use cases are illustrated in the figure below.

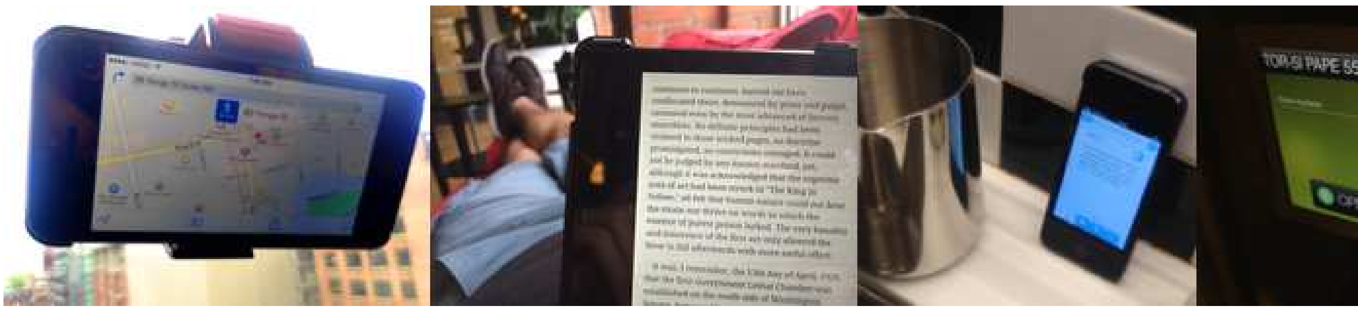


Fig. 1 Examples of tasks and contexts in which it is necessary to prevent the screen from going to sleep: from right to left, navigation from one location to another, reading an e-book, preparing a meal by following a recipe, and using a touch-enabled device as signage. In this final case, the device displays if a particular meeting room is booked - and allows users to manage booking a room by interacting with the screen, which is always on.

2.2 Keeping the system awake

On the flip-side, keeping the screen on is not the only kind of "wake lock" applications require. There is also use cases that there needs to be a kind of lock that also keeps the system awake for a short amount of time to allow arbitrary tasks to complete. It is well-known that, under normal conditions, the display of a device consumes a high percentage of a device's battery. Only keeping the screen on when absolutely necessary can greatly assist in prolonging the battery life of a device throughout a day.

As mobile devices are generally not as powerful as desktop class machines, and because of the uncertain network conditions under which these devices operate, some tasks can take a long time to complete - longer than users are willing to wait before putting their device back into their pocket, or sometimes longer than the default timeout most devices have to put the screen to sleep (e.g., 1 minute or there abouts). Consider importing and processing a set of contacts over a slow network connection, which can sometimes take minutes. It would be unrealistic to expect the user to hold their phone in their hand while they wait for a synchronization task to complete: users should be able to switch off the screen of a device and trust that an application will do its best to complete a certain task in the background.

In other words the user shutting off the screen doesn't mean that an application must stop running and have the device enter a low-power state. An application may need to intervene and finish what it's doing or it risks data-loss. In another concrete example a user may compose an email with a large photo attachment. When they press send they just as quickly press the power button to turn off the screen and put the phone back in their pocket. In such a situation, as happens with iOS's mail app, the application continues to run by sending the email before putting the phone in a low-power state. Note that this system level lock includes keeping the WiFi and cellular radio from also shutting down or entering a low power state while some task is being performed.

2.3 Potential for abuse

As with any device API on the Web, there is the risk that this API could be intentionally or unintentionally abused by developers.

Abuse cases include, but are not limited to:

- Requesting a wake lock on the system and/or the screen and then not releasing it. For the screen wake lock, if the user does not themselves power off the screen, this would cause the user's battery to be drained more quickly than normal. Likewise with the system lock.
- Requesting a system lock, and then exploiting it to perform computationally expensive operations without the user's consent while the screen is off (e.g., mining crypto-currencies and sending results to a server).
- Combining a system lock with an API like Geolocation could allow an application to track a user while the screen is off: as the system is not actually powered-down, it could continue sending information to a server without the user's knowledge.

Ultimately, it will be up to user agents to mitigate these sorts of abuse cases. However, the design of the API may also assist in mitigating certain kinds of abuses.

2.4 Current workarounds

To overcome the lack of support for wake locks in the platform, there is hearsay evidence that …Developers are resorting to hacks like playing hidden videos to prevent the screen from sleeping.

3. Methodology and limitations

We used convenience sampling to select the applications in this document. That is, we selected applications that we knew prevented the screen from going to sleep when the application was being used to perform some particular action. We also reached out to people on twitter for suggestions as to which applications we should look at. Suggestions were captured as "issues" in our GitHub repository.

As stated on Wikipedia's entry about convenience sampling: "[one] cannot scientifically make generalizations about the total population from this sample because it would not be representative enough." Regardless, we view the range of applications we collected as exemplifying common use cases for this kind of functionality. As such, we still created requirements from what we observed across applications and other platforms. We encourage the community to contribute more, or contradictory/abuse-cases, examples to the Web and Mobile IG. You can do this via a pull request to the screen-wake repository on GitHub.

3.1 How we looked at the apps

For each application, we asked:

- Why does the application need a wake lock?
- How does the user benefit?
- When is the wake lock applied and released?

4. Support on various platforms

Below is a list of APIs from native platforms that developers can make use of to prevent a device from going to sleep. The list shows that this functionality is commonly available to developers without significant security restrictions. It is important to note that some platforms differentiate between putting the system to sleep and putting the display to sleep. Requesting that the system (or "CPU") not sleep while allowing the display to sleep allows long running tasks to complete.

iOS:

UIApplication.idleTimerDisabled.

Android:

Wake locks: supports both CPU and display.

Windows 8

Idle detection for Windows Phone 8

FirefoxOS:

navigator.requestWakeLock() - Supports "cpu", "display", and "wifi".

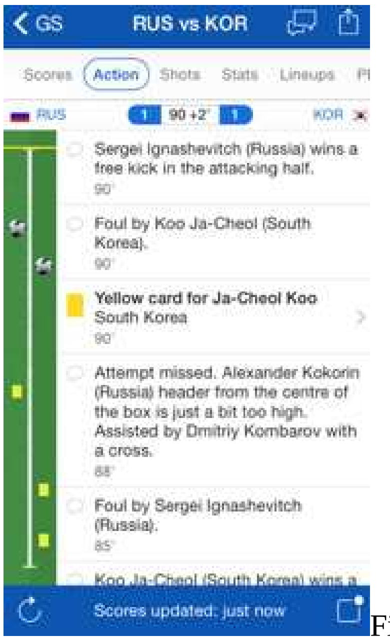
ChromeApps:

chrome.power.requestKeepAwake() - supporting both "system" and "display".

Tizen:

tizen.power.request(). Note that Tizen supports both "screen" and "cpu".

5. Brazil 2014


GS RUS vs KOR

Scores Action Shots Stats Lineups Pl

RUS 1 90 +2' 1 KOR

Sergei Ignashevitch (Russia) wins a free kick in the attacking half.
90'

Foul by Koo Ja-Cheol (South Korea).
90'

Yellow card for Ja-Cheol Koo
South Korea
90'

Attempt missed. Alexander Kokorin (Russia) header from the centre of the box is just a bit too high. Assisted by Dmitriy Kombarov with a cross.
88'

Foul by Sergei Ignashevitch (Russia).
85'

Koo Ja-Cheol (South Korea) wins a

Scores updated: just now

Fig. 2 The "Brazil 2014" iOS app allows users to keep up with news related to the 2014 World Cup.

Why does the application need a wake lock?

The application allows users to follow football matches in real-time: receiving updates as the match is taking place. Allowing users to follow the action in real-time seems to be the primary motivator.

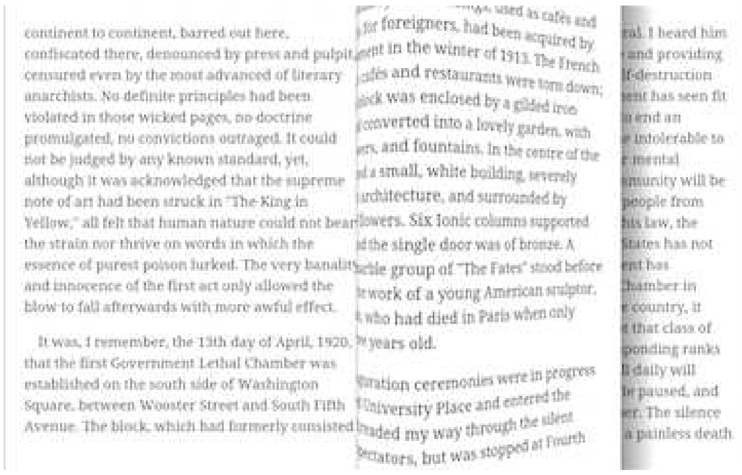
How does the user benefit?

During a match, the user can simply place their device on a surface for convenient viewing - or they can simply hold the device in their hand. Without requiring any user interaction light-weight textual and graphical information is streamed to the device and automatically displayed on screen. This means that the user does not need to interact directly with the device to receive updates of what is happening during a match. Note that this application does not stream live video.

When is the wake lock applied and released?

The wake lock is only applied when the user opens the "action" screen of the application and if, and only if, a match is actually taking place. The wake lock is released as soon as match is finished.

6. Google Play Books


continent to continent, barred out here, confiscated there, denounced by press and pulpit, censured even by the most advanced of literary anarchists. No definite principles had been violated in those wicked pages, no doctrine promulgated, no convictions outraged. It could not be judged by any known standard, yet, although it was acknowledged that the supreme note of art had been struck in "The King in Yellow," all felt that human nature could not bear the strain nor thrive on words in which the essence of purest poison lurked. The very banality and innocence of the first act only allowed the blow to fall afterwards with more awful effect.

It was, I remember, the 13th day of April, 1920, that the first Government Lethal Chamber was established on the south side of Washington Square, between Wooster Street and South Fifth Avenue. The block, which had formerly consisted

used as cafés and
for foreigners, had been acquired by
ent in the winter of 1913. The French
cafés and restaurants were torn down;
ock was enclosed by a gilded iron
converted into a lovely garden, with
ers, and fountains. In the centre of the
d a small, white building, severely
architecture, and surrounded by
flowers. Six Ionic columns supported
d the single door was of bronze. A
able group of "The Fates" stood before
e work of a young American sculptor,
who had died in Paris when only
years old.

ration ceremonies were in progress
University Place and entered the
aded my way through the silent
ectators, but was stopped at Fourth

ral. I heard him
and providing
lf-destruction
ent has seen fit
n end an
e intolerable to
r mental
munity will be
people from
his law, the
States has not
ent has
hamber in
e country, it
t that class of
ponding ranks
ll daily will
e paused, and
er. The silence
a painless death

Fig. 3 Books on Google Play is an Android application for managing and reading a collection of e-books.

Why does the application need a wake lock?

The Screen stays 'awake' allowing users to continue reading and e-book without having to continuously poke at the screen to keep the device awake.

How does the user benefit?

Allows a user the read at their own pace without needing to constantly poke the screen to keep it awake. It also means that users can keep their hands away from the screen until they need to turn the page.

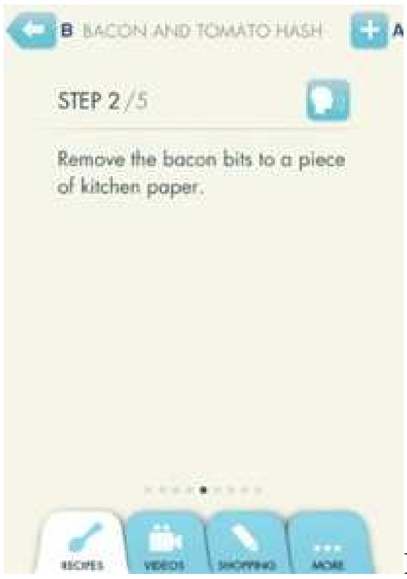
When is the wake lock applied and released?

The lock is applied only once the user opens a book. It is released once the user returns to the book selection menu. Additionally, if the user doesn't turn the page of a book after about 5 minutes, the application automatically releases the wake lock.

Applications that exhibit similar behavior:

- FBReader on Android is configurable. It default is to only prevent sleep when battery level is greater than 50%.

7. Nigela's quick collections



Remove the bacon bits to a piece of kitchen paper.

Fig. 4 Nigela's quick collections is a cooking application that contains a collection of recipes and a set of steps for how to prepare each dish.

Why does the application need a wake lock?

> Allowing users to prepare dishes while simultaneously interacting with the recipes of the application (recipes are spread across multiple screens and require multiple steps to be followed in order). A user may find it annoying if the device went to sleep while they were trying to follow a recipe (as this involves a lot of multitasking on the side of the user, their hands might not be available, or their hands might be dirty or wet).

How does the user benefit?

> As the user's hands could be dirty or wet, the app keeps the screen on and allows the user to navigate the application using voice commands instead.

When is the wake lock applied and released?

> The wake lock is only engaged when the user explicitly opens up a recipe. When the user leaves a recipe and returns to, say, the main screen, the wake lock is released.

Other applications that exhibit similar behavior:

- Allthecooks Recipes.

8. Google Maps

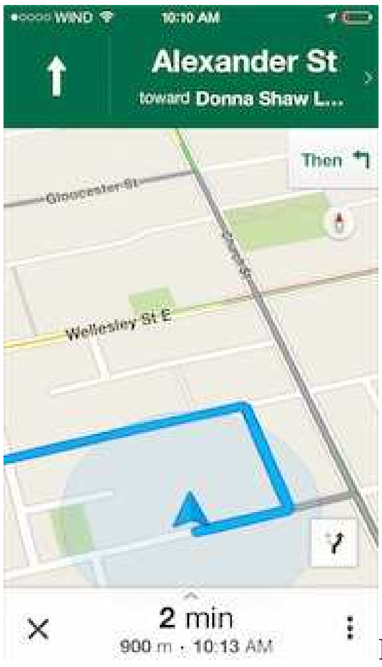


Fig. 5 Google maps on iOS is a popular maps application that supports turn by turn navigation.

Why does the application need a wake lock?

Safety. Keeping the screen on allows the user to periodically glance at their phone to get oriented while navigating to their destination. On iOS, even if the user turns off the screen, the application continues to run and notifies the when they need to make a turn.

How does the user benefit?

When in navigation mode, the screen will stay 'awake' so users do not need to physically interact with the device when navigating.

When is the wake lock applied and released?

The screen lock is only applied when the user begins navigating. The screen lock is released when the user stops navigating or reaches their destination.

Other applications that exhibit similar behavior:

- iOS Maps.

9. Xee Photo Viewer

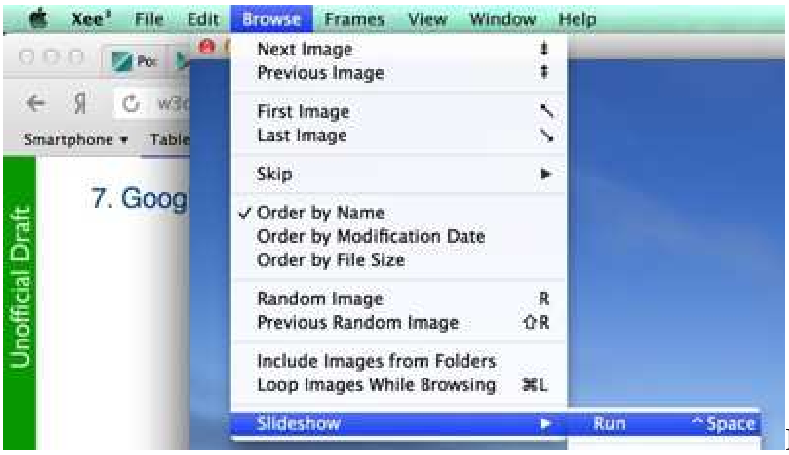


Fig. 6 Xee is an application for viewing images. It provides a "slideshow" mode, which automatically presents a sequence of images in full screen.

Why does the application need a wake lock?

> The wake lock makes sure that the screen does not switch off in the middle of the slideshow.

How does the user benefit?

> Becasue the screen stays 'awake', users do not need to physically interact with the device to ensure the slideshow continues running. This leaves the user free to interact with others whom may also be viewing the images in the same physical space.

When is the wake lock applied and released?

> The screen lock is only applied when the user starts the slideshow. The wake lock is released when the slideshow reaches the last photo in a set or it gets interrupted by the user.

Other applications that exhibit similar behavior:

- FastStone Image Viewer for Windows
- iPhoto

10. Firefox OS Contacts

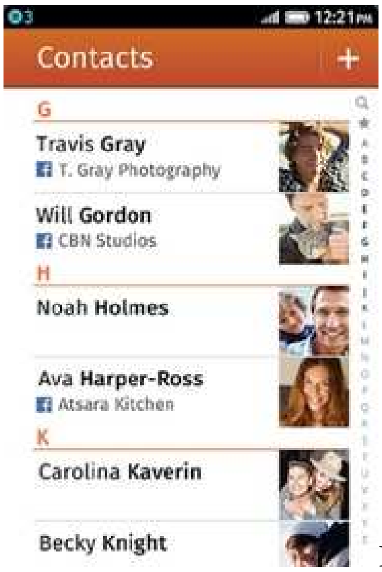


Fig. 7 Firefox OS's contacts application allows a user to manage contacts, as well as import friends from Facebook as contacts.

Why does the application need a wake lock?

To be able to import contacts and performing long-lived tasks without requiring the user to keep the screen on.

How does the user benefit?

The user is assured that importing contacts will complete without needing to keep the screen on. This conserves battery.

When is the wake lock applied and released?

When the import task is complete.

11. Other applications

A range of other applications benefit, or could benefit, from a wake lock. For instance:

- Google Cardboard needs to stay awake since there's no direct user interaction with the screen. It would benefit from having wake-lock functionality. Mozilla's VR efforts, which rely on the Oculus Rift, would also benefit from having this API.
- The "AliveECG - Heart Monitor" for Android is an application that monitors single-channel electrocardiogram (ECG) rhythm strips. The screen that shows the beats per minute prevents the screen from locking. This allows patients and medical professionals to monitor the ECG input in real-time without needing to worry that the device will go to sleep.
- Magic The Gathering Counter stays 'awake' for users to count lives without having the screen dim while playing.
- iOS's compass application uses a wake lock to prevent the screen from going to sleep. This allows users to orient themselves without needing to worry that the screen will power off unexpectedly.

12. Requirements

Any specification seeking to address these *requirements*:

1. *MUST* support requesting a wake lock on at least the **system** and on the **display**. The display being the screen or primary output modality of the device. The system meaning that the device can put the screen to sleep, but *MUST* continue running a task until an application releases the lock on the system. Keeping the screen awake *MUST* (obviously) also keep the system awake.
2. *MUST* be extensible, allowing for new types of locks to be specified in the future.
3. *MUST* support a way for user agents to support wake locks only under specific security conditions (e.g., only in full screen).
4. *MUST* support a means for scripts to be notified if a request for a lock fails to be granted. And if a lock request is not granted, the UA *SHOULD* provide the developer and script with appropriate information detailing why the request was not granted.
5. *MUST* allow developers to release a previously requested wake lock.
6. *MAY* allow scripts to check if an origin already holds a particular kind of lock.
7. *MUST* make the user agent ultimately responsible for managing the wake locks. The user agent *MAY* automatically release the lock for security reasons, and *MUST* release all locks when an application is closed or navigated to another origin. It *MAY* also release the lock after some amount of inactivity.
8. *MUST NOT* be applied solely at the application level (e.g., just once when the application starts). Instead, developers *MUST* be able to request wake locks at arbitrary moments in time to help the user complete a task.
9. *MUST* be possible for the end user to override the wake lock on the screen (e.g., by pressing the power button on the device, which puts the screen to sleep). Applications *SHOULD* be notified when such a thing happens, so they can request a system lock if they need to complete some task.
10. *MAY* allow a means to hint to the user agent how long a lock is to be held (e.g., "keep the screen on for at least 5 mins").
11. *MUST* allow the end user to have ultimate control over which documents have control over a wake locks (e.g., not allowing a page to get get any kind of wake lock without user opt-in).

A. Observations

The following are a set of related observation we made while we were preparing this document.

In iOS, an orientation change can serve as a trigger to prevent the screen from locking. However, this only works on apps that support changes in orientation. This is not the case in Android, where changing the orientation of the device has no effect - even if the application supports multiple orientations.

B. Acknowledgments

Huge thanks to Ehsan Akhgari, Seth Ladd, Ian Hickson, Boris Smus, Damon Douglas, Ilya Bogdanovich, Tab Atkins, and Domenic Denicola.

Appendix 3: Matrix of Permissions Usage in the Web Platform

The research on the state of permissions as they are currently managed on the Web is summarized in the matrix below.

Feature	Type of permission request	Time of request	Persistence of permission	Visibility of permission	Control on permission	Embeddability	Permission d
Geolocation	Prompt	At usage time	Session by default, optionally permanent	Chrome indicator	Via indicator	Always allowed, bound to origin of document calling script	Error callback `PERMISSION_D`
Direct camera access (`getUserMedia`)	Prompt	At usage time	Session by default, optionally permanent	Chrome indicator, pulsing reminder	Via indicator	Always allowed, bound to origin of document calling script	Error callback `PermissionDe`
Direct mike access (`getUserMedia`)	Prompt	At usage time	Session by default, optionally permanent	Chrome indicator	Via indicator	Always allowed, bound to origin of document calling script	Error callback `PermissionDe`
Get stuff from camera / mike (HTML Media Capture)	Implict via user interaction	At usage time	One-off	N/A	N/A	Always allowed	No file object a
Notification	Prompt	Upfront	Persistent	None (?)	?	Allowed	`Notification` `"denied"`
Pop Up	Denied by default, Opt-in required	At usage time	One-off, optionally persistent	Chrome indicator	Via indicator	Allowed	`window.open`
Fullscreen	Ask for forgiveness after “engagement gesture”	At usage time	One-off	Transient overlay	Through instructions	Allowed only with specific permission from parent (allowfullscreen attribute)	watch `el.full` `fullscreener` previously reje
Pointer Lock	Ask for forgiveness after “engagement	At usage time	One-off	Transient overlay	Through instructions	Allowed	`document.poi` `== null`; `poin` event

Feature	Type of permission request	Time of request	Persistence of permission	Visibility of permission	Control on permission	Embeddability	Permission c
	gesture”						
File picker	Implicit via user interaction	At usage time	One-off	N/A	N/A	Allowed	No file object
Quota	Prompt	Upfront	Persistent	?	Convoluted	Allowed	Success (!) call `StorageInfo.`