HTML5Apps

Deliverable D3.2

"Apps" Standards Coordination report 2

9 November 2015

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:	D3.2
Deliverable title	“Apps” Standards Coordination report 2
Contractual Date of Delivery:	M24
Actual Date of Delivery:	9 November 2015
Author (s):	Daniel Dardailler
Reviewer (s):	Dominique Hazäel-Massieux , Dr. Philipp Hoschka
Work package no.:	3
Work package title:	Standards Coordination
Work package leader:	Daniel Dardailler
Distribution:	PU
Version/Revision:	1
Draft/Final:	Final
Total number of pages (including cover):	49
Keywords:	Concertation meeting, Open Web Platform

Disclaimer

This document contains description of the HTML5Apps project work and findings.

The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the HTML5Apps consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 27 Member States of the Union. It is based on the European Communities and the member states cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors. (http://europa.eu.int/)

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

History

Version	Date	Comment
0.1	21/09/2015	First draft report
1.0	9/11/2015	Final version

TABLE OF CONTENT

1. INTRODUCTION 6

2. OUTREACH CAMPAIGN 7

2.1. Approach 7

2.2. Project Feedback 8

2.3. Conclusion 10

3. WEB AND CLOUD ROADMAP 11

3.1. Core Web Design and Development (Graphics and Layout, Device Adaptation, Forms, Data storage) 11

3.2. Media and Real-Time Communications 12

3.3. Usability and Accessibility 12

3.4. Device Interaction 12

3.5. Network Integration 12

3.6. Application Lifecyle 12

3.7. Payment and Services 13

3.8. Performance & Tuning 13

3.9. Security & Privacy 13

3.10. Web of Things 13

3.11. Summary 13

4. CONCLUSION 15

ANNEX A 16

ANNEX B 21

1. INTRODUCTION

The mission of the HTML5Apps work on standardisation coordination was to engage R+D projects in the area of "Software Engineering, Services and Cloud computing" in the development and adoption of HTML5 standards.

In the first year of the project, the standards coordination efforts of the HTML5Apps project has studied and talked to the projects active in the area of Software Engineering Services and Cloud Computing to (1) find out which Standards Developing Organizations (SDOs) are relevant to them and (2) help the projects understand the openness level of the various SDOs they are basing their innovation on (e.g. OpenStand principles).

As recommended by the European Commission, in the second year, we focussed our standards coordination effort on engaging relevant European stakeholders and building a community to support development and adoption of HTML5 standards.

Accordingly, we identified a subset of projects in both FP7 and H2020 call 1 that could have an interest in the development and adoption of HTML5 standards. We put a particular focus on PAAS (Platform as a Service) projects and projects that defined APIs.

Analysis of and outreach to the projects revealed that a significant number of HTML5 standards have the potential to play an important role in Cloud technology, but this is not widely known due to a "communication gap" between the Web and the Cloud communities.. .

To address this, HTML5Apps developed a specialised version of the general HTML5Apps roadmap (D3.4) focussing on HTML5 standards that are of particular relevance to Cloud Computing research and development. This roadmap allows Cloud-related projects to more easily keep track of and participate in ongoing W3C work..

An initial version of the roadmap was presented and well received at the "Towards secure and trusted cloud services in Europe" workshop held at the European Commission on September 24th.

This report is structured as follows: In Section 2, we describe our outreach on HTML5 standards to individual R&D projects in the area of "Software Engineering, Services and Cloud computing", and the results of this outreach. In Section 3, we describe the Web and Cloud roadmap we developed. Section 4 concludes the report.

2. OUTREACH CAMPAIGN

2.1. Approach

We categorized projects in the "Software Engineering, Services and Cloud computing" area into three categories:

- Platform/API development oriented projects, Cloud or else, FP7 or H2020

  BETaas, BigFoot (Data), CloudSpace, Clout (IoT), CoherentPAAS, COMPOSE (IoT), MODAClouds, OPENi, PAASAGE, SeaClouds, ESCUDO (H2020, data), PAASWord (H2020, data privacy)
- Methodology/QA oriented projects

  E.g. ARTIST, BrokerCloud, GENIC, etc.
- Misc/Outreach/PSI pilot oriented projects, IAAS

  E.g. StormCloud, STRATEGIC, CLIPS, etc.

The HTML5Apps project interest focuses on Platform/API developers, that is, projects in the first list.

Messaging:

Based on previous unsatisfactory results with general outreach to get projects attention, we decided to research all projects' Web site to find the right technical, or even better, standardization manager, and to send a personal message to each of them, showing that we had taken the time to understand their technical work.

We send about 15 specifically tailored messages to the technical contacts identified in different projects,

Here is an example of dialog to obtain information on a given project interest in HTML5Apps work.

```
-------- Original Message --------
Subject: MODAClouds and Web APIs
Date: Mon, 20 Apr 2015 11:21:55 +0200
From: Daniel Dardailler <danield@w3.org>
Organisation: W3C
To: petcu@info.uvt.ro


Hello Dana

As you probably know, W3C is working on extending its Open Web Platform, that
is to say, the set of APIs available to Web developers in a browser
environment, such as HTML5 (and it's JS interfaces).

As part of our HTML5Apps EC Project, we're looking for potential APIs
development done in other Unit projects, around Cloud, IoT, Open data,
Privacy, etc., that would allow access to their services provided from within
a Web page.
```

```
From a cursory look, MODAClouds sounds like a good candidate, since you are
defining a Platform and associated APIs for use by programmers, but I was
wondering if you have looked at this specific aspect (i.e. extending HTML5).

Thanks for your input in any case.
```

And here the answer received:

```
-------- Original Message --------
Subject: Re: MODAClouds and Web APIs
Date: 2015-04-26 09:23
From: Dana Petcu <petcu@info.uvt.ro>
To: Daniel Dardailler <danield@w3.org>

Dear Daniel

We have investigate inside the consortium  and only one component,
models@runtime, can claim a connection.
Its webpage uses the websocket protocol (potential still part of HTML5
specification).
The contact person for this component is Nicolas Ferry
<Nicolas.Ferry@sintef.no>

Best regards
Dana Petcu
```

Which would then allow us to then contact the standard expert in the project and get the right requirements for building our standard for Cloud specific roadmap.

We received feedback from six projects over the course of a few weeks, which is presented in the next section.

2.2. Project Feedback

Based on the personal messages send to all contacts in the PAAS oriented project list, we received feedback from the following six projects:

PAASAGE

Project concerned with Model-based Cloud Platform Upperware. We contacted Keith Jeffery, scientific coordinator.

Two meetings occurred as a result of this outreach. We presented the W3C HTML5 work to the technical team (PEB).

PAASAGE is mostly interested in the user

		interface aspects provided by the Web in the model architecture they use. Several scenarios were used for illustration. E.g. an HTML5 document could use some API to call PaaSage; the author could indicate that the reader can 'try out' the CLOUD deployment of an application (for example in automotive engineering) used to produce the scientific conclusions in the paper. Similarly for an economist writing an article in a professional magazine (or even a newspaper) about predictions of economic growth by country / by sector and providing the API to PaaSage so people could try out the application deployment to see if they agreed with the conclusions.
	PAASWORD	Project focuses on secure storage of both corporate and personal sensitive data on Cloud infrastructures. Initial contact with Simone Braun, discussing potential ontology standards. Further conversations did not bring more cooperation with HTML5Apps since APIs are not a focus of the project.
	MODAClouds	Project focus is on design and execution of applications in multiple clouds. One component, models@runtime, uses the websocket protocol. Contact person for this component is Nicolas Ferry. The websocket work at W3C is part of our roadmap, in the Local network services. The importance of this specification for the cloud is a valuable input for our custom Cloud roadmap.
	CloudSpaces	Project targeting the sharing of information between heterogeneous Personal Clouds. Pedro Garcia Lopez reported that CloudSpaces was finishing interoperability tests between APIs. During our exchanges, we studied their Interoperability protocol[1] which enables different Personal Clouds to share resources among them via an API, without forcing users to be in the same provider. This is interesting as a way to test interoperability

[1] https://github.com/cloudspaces/interop-protocol

		of mobile apps with cloud capacities, and compare with the compatibility of pure Web apps.
	COMPOSE	Focus on Web Of Things, W3C is already involved with Dave Raggett, also working on HTML5Apps, being the scientific contact. A new standardization activity is already on its way in this area.

We also received a reply from the H2020 **ESCUDO** project, where Pierangela Samarati replied saying that they had not looked at Cloud API standardisation yet but since the project was just beginning there might be later opportunity for providing ESCUDO-CLOUD APIs.

2.3. Conclusion

A common thread in our conversations with projects was that several of the core HTML5 standards were of interest, or in use by projects already, but there was a lack of understanding of how the projects could use them (or use them better), depending on their degree of maturity, implementation, or other deployment criteria.

To help orient the Cloud community in the area of HTML5 standardisation, we therefore decided to produce a custom version of the “Apps” standardisation roadmap pointing out which areas are of particular relevance to Cloud computing, and which are less relevant.

3. WEB AND CLOUD ROADMAP

The HTML5Apps project aims to **educate** the R&D community involved in the Software and Cloud areas - where the active SDOs are legions - on the importance of the Open Web Standardisation for the Cloud layers.

Dialogs with projects during this reporting period focused on trying to understand their interest in producing Cloud oriented APIs (e.g. related to privacy, choreography, modelling, etc) that could be added to the Open Web Platform through new HTML5 APIs, which the core of HTML5Apps is concerned with.

The "Standards for Web Applications on mobile **roadmap"**[2] (D3.4) is a deliverable of our project that informs all Web programmers, on a quarterly basis, of the evolution of Web standards particularly relevant to mobile.

This document aims at facilitating the adoption of the Open Web Platform as a target developer platform for ICT research projects among others.

The document gives information about status, deployment, etc. of different HTML5-application standards in development at W3C.

Given the focus of the "Software Engineering, Services and Cloud Computing" projects, on Cloud, we developed a version of our generic roadmap highlighting HTML5 standards work that is relevant for Cloud computing, which can be found in Annex B.

The analysis was also presented at the Workshop entitled "Towards secure and trusted cloud services in Europe" on September 24th.

The online version of the Cloud Web roadmap was promoted through the CloudWatch distribution list, concertation@cloudwatchhub.eu, and send, in addition to the Mobile roadmap, to Anja Köhler, in charge of the new European Future Internet Portal[3], to bring this additional data to that portal as well.

In the following section, we look at each of the foundations described in the roadmap of HTML5 standards and describe their relevance with respect to Cloud computing

3.1. Core Web Design and Development (Graphics and Layout, Device Adaptation, Forms, Data storage)

The Graphics and Layout layers are not very relevant for the Cloud programmers, they are part of the UI considerations. That being said, the Web provides a valuable portable layer for Cloud application UIs, allowing Cloud researchers to concentrate on the PAAS/IAAS level.

W3C work on IndexedDB and background synchronisation create a good combination needed for Cloud storage so it is something Cloud designers should track.

[2] http://www.w3.org/Mobile/mobile-web-app-state/

[3] http://future-internet.eu

3.2. Media and Real-Time Communications

Increasingly, sharing/streaming media is a big use case for cloud technologies, as the cloud makes everything faster and appear closer on the net, large binary objects in particular.

The natural distribution of media on a given Web page, coming from different servers, in different authenticated streams, should lead to a Cloud friendly architecture but Cloud designers are not always at the table in HTML5 standardisation to raise their requirements.

3.3. Usability and Accessibility

Although focused on UI as well, Cloud programmers should pay attention to MMI, the multimodal interface activiities of W3C, since in addition to integrating multiple UI modalities, e.g., GUI, speech, touch and gesture, the scope of the latest charter of the multimodal interaction Working Group includes combination of cloud services and multiple input/output modalities provided by more than one devices.

For more details, please see the MMI Ecosystem report.

3.4. Device Interaction

A primary use case for Cloud technologies in the near future will be to handle data gathered from the myriad of sensors that get build and distributed in devices all over the planet.

Web technologies can increasingly be used to interact with these sensors.

3.5. Network Integration

All work in this area is relevant: synchronisation, push, socket, XMLHttpRequest, are all used in cloud agents and need to be tracked by cloud R+D projects for new features.

3.6. Application Lifecyle

While Cloud services are potentially always in operation, their usage by end-users depend on their proper integration in the clients that they interact with, whose lifecycles depend on many parameters: battery, network connectivity, visibility on the device, etc.

These notions are part of the overall *application lifecycle*: how applications get installed, shown to the user in applications list, started, stopped, woken up from remote notifications, synced up when the device goes on-line.

These various capabilities are brought the Web platform through different mechanisms such as Packaging on the Web or the JSON-based manifest format and the HTML5's ApplicationCache work.

3.7. Payment and Services

HTML5App's new W3C activity on payment is already looking at Cloud integration, e.g. differences between eWallets that reside in your phone or in the cloud, or more generally any payment card details managed either on a secure element or on the cloud. Of course, the things people buy online, the actual data or resource may be outsourced to a cloud service provider and so communication and protocols must be developed in this context.

3.8. Performance & Tuning

Concerns mostly the Web itself, but the cloud being a performance driver as well, there may be connections to make.

3.9. Security & Privacy

Clearly a big intersection with the Cloud, and all Cloud programmers should follow this work if they want to write secure cloud web apps, concerned with identity, encryption, etc.

3.10. Web of Things

The **Web of Things**, a new W3C activity, is also very relevant, as it focuses on servers ranging from micro-controllers to cloud based server farms where large numbers of sensors, high message through put and big data are very much to the fore. W3C's contribution in this area focuses on metadata as an enabler to implementing an abstraction layer that sits above the platforms and protocols, a bit like the Web itself sits on top of lower level Internet protocols.

Also worth mentioning, most of the Cloud Computing API work is based on URIs and REST, concepts developed by the W3C and IETF, so these needs to be tracked as well. And so does our recent work on Efficient XML, EXI, which is used by a lot of frameworks for exchanging structured data.

3.11. Summary

In developing this new roadmap resource, we aim at lowering the risk that PaaS-based Web applications restrict overall Web application portability. There is an opportunity for the W3C to utilise its experience in standardising Web technology to aid PaaS web application portability and define how Web applications can be structured to utilise cloud resources. There are a number of existing development efforts within the W3C which could be leveraged and we hope that researchers will take these opportunities to strengthen their platform.

Overall, we see two core issues which currently limit PaaS application portability and that we hope using the Web platform can help solve.

First, the variety of model abstractions for traditional cloud resources makes it hard for developers to create applications which can port easily between PaaS providers.

Second, even though using Web technologies in a pervasive way, e.g. URI and REST, or HTML5, looks like a guarantee of compatibility between platforms, the approach

taken by Cloud environments does not provide application portability for PaaS and developers still have to re-implement their application code in a different development language to move between PaaS providers supporting different languages.

4. CONCLUSION

For the Cloud technology to be open and interoperable across Europe, but also across the entire Internet, it has to rest on other open standards.

Cloud applications that use native APIs, proprietary to a particular vendor, will fail to deliver true portability.

Our objective in HTML5Apps to "**close the gap**" between Open Web apps and native one-platform-only apps, is therefore tightly aligned with the Cloud standardization objectives of being open and pervasive.

To achieve this objective, we have conducted outreach and coordination on both sides of the landscape (research and standards).

We have focused our attention to projects defining APIs for the cloud (PAAS layer usually), and on ways to help them consider the HTML5 and Open Web Platform as a natural complementary to making PAAS software and standards as open as the Web.

As a result, we have produced a version of our HTML5/Web standard roadmap with a specific view on Cloud relevance, to raise awareness within the Cloud R&D community on the importance of the Web platform for their projects.

ANNEX A

This slide set was presented by Daniel Dardailler at the workshop "***Towards secure and trusted cloud services in Europe***" on September 24, 2015 in Brussels.

W3C Roadmap and Cloud

Towards secure and trusted cloud services in Europe
September 24, 2015 in Brussels.

html5apps-project.eu

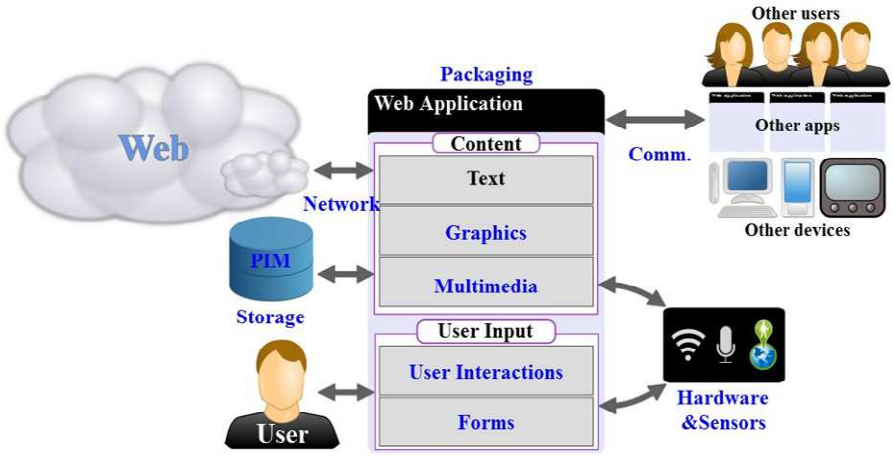
Sept 2015 | **Dr. Daniel Dardailler - W3C**

World Wide Web Consortium

1

- Web Universality
- Founded by Tim Berners-Lee in 1994
- W3C Standards: HTML, CSS, XML, WAI, RDF, *http/url/rtc*
- About 80 staff, 4 hosts, 40% Europe

The Open Web Platform

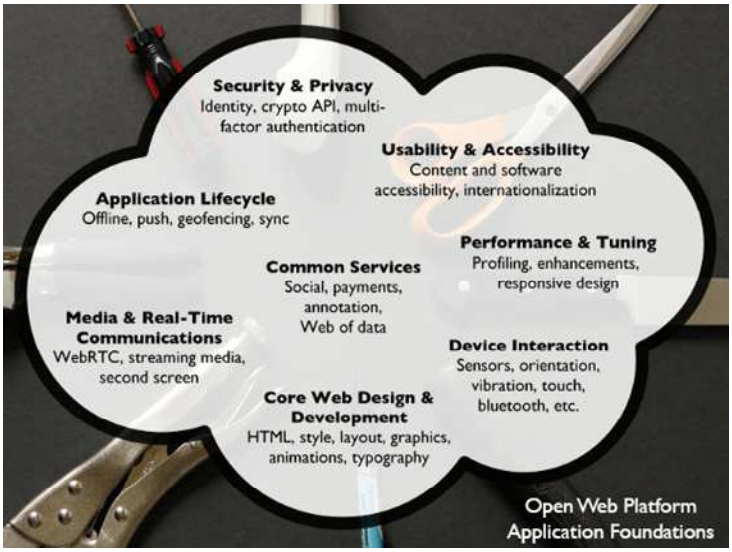


Web and Cloud

- Cloud as a specialized application of the Web
  - use URIs, HTTPS, XML, Web Services, etc.
  *Need for more **Secure** Web Standards*

- Web apps as generic consumer of Cloud services
  - Storage, Data-intensive, integration WoT
  Need for new Interoperable Standards

Application Foundations



The Roadmap

- Based on Application Foundations

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Simple data storage	Web Storage	Web Applications	Rec	Finished	Finished	Well deployed	WebPlatform.org	Complete
	File API		WD	Stabilizing	Last updated April 2015	Getting well deployed	WEBDEV CAMPUS	Started
File operations	FileSystem API		N/A	Early proposal	Last updated July 2015	N/A		None

Cloud relevance

7

- Highly relevant:
  - Security/Privacy (identity, signature, encryption)
  - Web Payments (wallet, API, Card details)
  - Network Integration (x-origin, socket, rtc, etc)
- Relevant:
  - Web of Things
  - MMI
  - Data storage
- Less relevant: UI, App cycle, perf, etc

In Summary

The Web is the main entry point for online end-users.

Similar issues: Security, Privacy, Usability, etc.

Lessons from Web standardization:

- Open Participation, coordination
- Royalty-free standards, Open Source
- Coherent Architecture

ANNEX B

Annex B represents the subset of our generic W3C HTML5Apps roadmap highlighting standard work that is relevant to EU R&D projects developing Cloud software, in particular for the PAAS and SAAS layer oriented projects.

It is attached as PDF to this report and can be found on our Project Web site as well: https://www.w3.org/2015/09/HTMLApps-D3.4/cloud.html

Cloud Services and Standards for Web Applications: current state and roadmap

This document summarizes how technologies currently developed in W3C apply to the Cloud context. This is a subset of our generic HTML5Apps roadmap highlighting standard work that is relevant to EU R&D projects developing Cloud software, in particular at the PAAS and SAAS layers.

1. Core Web Design and Development....3
2. Media and Real-Time Communications....5
3. Usability and Accessibility....9
4. Device Interaction....9
5. Network Integration....13
6. Application Lifecyle....17
7. Payment and Services....20
8. Performance & Tuning....22
9. Security & Privacy....25

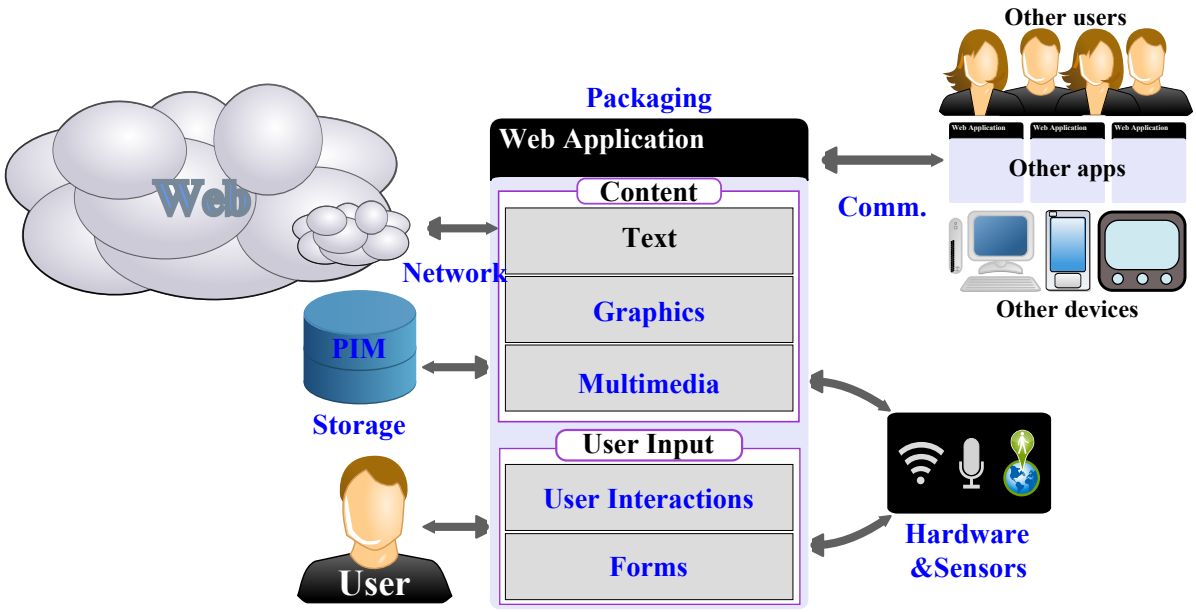
Document structure

Features in this roadmap are organized around the application foundations for the Open Web Platform, a set of high-level components that application developers rely on to build their Web-based content and services.

The following application foundations are considered in this document: core web design and development (page 3), media and real-time communications (page 5), usability and accessibility (page 9), device interaction (page 9), application lifecycle (page 17), payment and services (page 20), performance & tuning (page 22), and security & privacy (page 25). In addition, it covers topics related to network integration (page 13).

Beyond the areas covered below, the following W3C areas are relevant for Cloud services:

- the W3C Web of Things activity is also very relevant, as it focuses on servers ranging from micro-controllers to cloud based server farms where large numbers of sensors, high message through put and big data are very much to the fore. W3C's contribution in this area focuses on metadata as an enabler to implementing an abstraction layer that sits above the platforms and protocols, a bit like the Web itself sits on top of lower level Internet protocols.
- most of the Could computing API work is based URIs and REST, concepts developed by the W3C and IETF, so these needs to be tracked as well.
- Efficient XML (EXI) is used by a lot of frameworks for exchanging structured data.
- Cloud programmers should pay attention to MMI, the multimodal interface activtities of W3C, since in addition to integrating multiple UI modalities, e.g., GUI, speech, touch and gesture, the scope of the latest charter of the multimodal interaction Working Group includes combination of cloud services and multiple input/output modalities provided by more than one devices.



The Web as an application development platform

In each category of features, a table summarizes for each feature:

- which W3C specification defines the feature,
- which W3C group is responsible of the said specification,
- the stage of the specification in the W3C Recommendation track (see below),
- the estimated stability of the feature, i.e. how little the author expects it to change, from an early draft that can still evolve a lot, to a finished document with only minor expected changes,
- a link to the latest editors draft of the document, and a representation of the recent editing activity;
- some qualitative indication on availability of implementations on mobile devices, based on data collected primarily from Can I Use… and mobile HTML5, completed with data from Mozilla developer network, QuirksMode, JWPlayer's state of HTML5 video, Chromium Dashboard, Internet Explorer Platform status, the Device APIs Working Group Implementation status as well as the author's understanding of the mobile devices market (see also the code used to generate the support icons)
- When available, a link to a relevant tutorial on WebPlatform Docs, and to relevant on-line training courses on W3DevCampus
- a link to the test suite for the said feature, and when relevant, a github ribbon to access the underlying git repository.

W3C creates Web standards by progressing documents through its Recommendation track, with the following stages:

"Editors drafts" represent the current view of the editors of the specification but have no standing in terms of standardization.

"Working Drafts" (WD) are early milestones of the Working Group progress.

"Last Call Working Drafts" signal that the Working Group has determined that the specification fulfills its requirements and all the known issues have been resolved, and thus requests feedback from the larger community.

"Candidate Recommendations" (CR) trigger a call for implementations where implementors are invited to implement the specification and send feedback; Working Groups are expected to show the specification gets implemented by running test suites they have developed.

"Proposed Recommendations" (PR) manifests that the group has gathered sufficient implementation experience, and triggers the final review by W3C Members

"W3C Recommendations" (Rec) are stable and completed Web standards; these documents only get updated rarely, through the "Edited Recommendation" process, as a results from errata collected by Working Groups.

For groups that have adopted it, the 2014 update of the W3C Process simplifies a bit the progression by removing the Last Call stage — instead of a single global call for review addressed to the whole community, Working Groups are empowered with solicitting reviews from their various related communities as long as they can demonstrate sufficient wide review of the specification before requesting transition to Candidate Recommendation.

Prior to starting standardization, a Working Group needs to be chartered, based on input from W3C Members, often through the organization of a workshop, or after the reception of a W3C Member Submission.

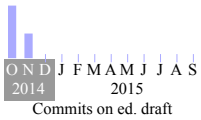
W3C has set up Community Groups, a mechanism that allows anyone to do experimental work within the W3C infrastructure, under IPR rules that are compatible to transition the work to the W3C standardization process.

1. Core Web Design and Development

Overall, he Graphics and Layout layers are not very relevant for the Cloud programmers, they are part of the UI considerations. That being said, the Web provides a valuable portable layers for Cloud application UIs, allowing them to concentrate on the lack of standards at the PAAS/IAAS level.

However, IndexedDB and background synchronisation create a good combination needed for Cloud storage so it is something Cloud designers should track.

Some of this data need to be encrypted, the Web Cryptography API from the Web Cryptography Working Group exposes strong cryptography primitives to Web applications, and can be bound to pre-provisioned keys via the WebCrypto Key Discovery API.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Cloud storage	Indexed Database API	Web Applications	Rec	Stable	Finished	Well deployed 8+ 4.4+ 44+ 30+ 10+ 40+	WebPlatform.org W3DEV CAMPUS	Good coverage
	Web Background Synchronization	Web Applications	N/A	Early draft	Last updated April 2015 	None X X X X X X		None
Encrypted storage	Web Cryptography API	Web Cryptography	CR	Stable	Last updated November 2014 	Well deployed 8+ 44+ 44+ 30+ 11+ 19+		Early start
	WebCrypto Key Discovery	Web Cryptography	WD	Early work	Last updated May 2014	None X X X X X X		None

2. Media and Real-Time Communications

More and more, sharing/streaming media is a big use case for cloud technologies, as the cloud makes everything faster and appear closer on the net, large binary objects in particular.

The natural distribution of media on a given Web page, coming from different servers, in different authenticated streams, should lead to a Cloud friendly architecture but Cloud designers are not always at the table to raise their requirements.

HTML5 adds two tags that dramatically improve the integration of multimedia content on the Web: the **`<video>`** and **`<audio>`** tags. Respectively, these tags allow embedding video and audio content, and make it possible for Web developers to interact much more freely with that content than they would through plug-ins. They make multimedia content first-class citizens of the Web, the same way images have been for the past 20 years.

The playback content can be streamed, augmented and completed via Media Source Extensions that lets developers buffer and generate media content in JavaScript.

To cater for the needs of some content providers, a proposal to enable **playback of protected content**, *Encrypted Media Extensions* is an API that is under consideration in the HTML Working Group.

While the new HTML5 tags allow to play multimedia content, the *HTML Media Capture* defines a **markup-based mechanism to access captured multimedia content** using attached camera and microphones, a very common feature on mobile devices. The Web Real-Time Communications Working Group and the Device APIs Working Group are building together an API (`getUserMedia`) to directly manipulate **streams from camera and microphones**, as well as an API to record these streams into files, and another API to use access to cameras to take photos programatically. This makes it easy for Cloud-based media processing content to obtain content from end-user devices.

Beyond capturing and recording, two additional APIs add multimedia manipulation capabilities to the Web platform. We have already mentioned the *Canvas 2D Context* API: it enables modifying images, which in turn opens up the possibility of **video editing**.

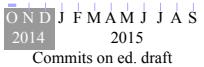
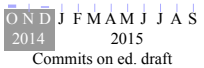
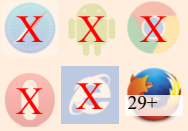
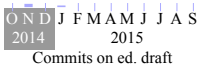
In a similar vein, the Audio Working Group is working on an API that that makes it possible to modify audio content, as well as **analyze, modify and synthesize sounds**, the Web Audio API.

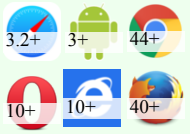
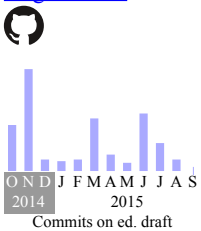
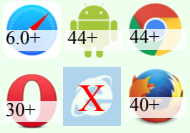
The **Web Real-Time Communications Working Group** is the host of specifications for a wider set of communication opportunities:

- **Peer-to-peer connection** across devices,
- **P2P Audio and video streams** allowing for real-time communications between users.

The combination of all these features marks the starting point of the Web as a comprehensive platform for multimedia, both for consuming and producing. The rising interest around bridging the Web and TV worlds (manifested through the W3C Web and TV Interest Group) should strengthen that trend in the coming months. Mobile devices are expected to take a growing role in many users TV experience, providing a “second screen” experience, where users can find more information on or interact with a TV program they're watching via their mobile devices.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Video playback	video element in HTML5	HTML	Rec	Stable	Finished	Good deployment 3.2+ 2.3+ 44+ 11+ 10+ 40+	WebPlatform.org W3DEV CAMPUS	Well started
Audio playback	audio element in HTML5	HTML	Rec	Stable	Finished	Good deployment 3.2+ 2.3+ 44+ 11+ 10+ 40+	WebPlatform.org W3DEV CAMPUS	Started
Generation of media content	Media Source Extensions	HTML	CR	Stable	Last updated July 2015 O N D 2014 J F M A M J J A S 2015 Commits on ed. draft	Well deployed X 4.4.3+ 44+ 30+ 11+ X	WebPlatform.org	Well started
Protected content playback	Encrypted Media Extensions	HTML	WD	Early draft	Last updated August 2015 O N D 2014 J F M A M J J A S 2015 Commits on ed. draft	Limited X X 34p+ X 11+ X		None

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Capturing audio/ video	HTML Media Capture	Device APIs		Stable	Last updated October 2014 	Growing deployment 		 Good coverage
	Media Capture and Streams	Device APIs and Web Real-Time Communications		Stabilizing	Last updated August 2015 	Growing 		 started
	MediaStream Recording	Device APIs and Web Real-Time Communications		Early draft	Last updated December 2014 	Very limited 		None
	Mediastream Image Capture	Device APIs and Web Real-Time Communications		Early draft	Last updated January 2015 	None 		None

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Image & Video analysis, modification	HTML Canvas 2D Context	HTML	CR	Stable	Last updated December 2014  Commits on ed. draft	Widely deployed 		Good coverage
Audio analysis, modification	Web Audio API	Audio	WD	Starting to stabilize	Last updated August 2015  Commits on ed. draft	Good deployment 		Started
P2P connections and audio/video streams	WebRTC 1.0: Real-time Communication Between Browsers	Web Real-Time Communications	WD	Early draft	Last updated June 2015  Commits on ed. draft	Growing 		Early start

3. Usability and Accessibility

UI considerations are not very relevant for the Cloud programmers, but the Web provides a valuable portable layer for cloud applications UIs.

4. Device interaction

A primary use case for Cloud technologies in the near future will be to handle data gathered from the myriad of sensors that get build and distributed in devices all over the planet.

Web technologies can increasingly be used to interact with these sensors.

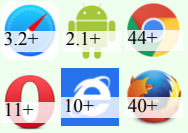
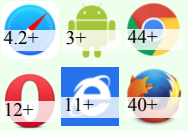
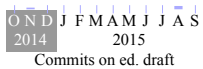
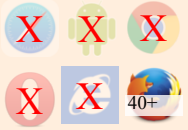
The ***Geolocation API*** provides a common interface for locating the device, independently of the underlying technology (GPS, WIFI networks identification, triangulation in cellular networks, etc.).

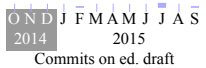
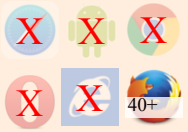
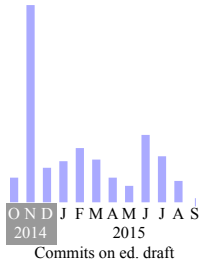
Web applications can also now access **orientation and acceleration** data via the *DeviceOrientation Event Specification*.

A number of APIs for other sensors are under development: the Battery Status API, the *Proximity Events* API, the *Ambient Light Events* API or the proposed *Ambient Humidity Events* API. The Device APIs Working Group has started an effort to propose a unification pattern for these various sensors.

As already mentioned in the section on multimedia (page 5), there is ongoing work on APIs to open up access to camera and microphone streams.

A Web Bluetooth Community Group was started to develop a Bluetooth API for browsers with a particular goal of supporting Bluetooth Low Energy devices.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Geolocation	Geolocation API Specification	Geolocation		Finished	Finished	Widely deployed 		 Good coverage
Motion sensors	DeviceOrientation Event Specification	Geolocation		Stabilizing, but with planned updates	Last updated August 2014 	Well deployed 		Started
Battery Status	Battery Status API	Device APIs		Stable	Last updated August 2015  Commits on ed. draft	Growing 		 Good coverage
Proximity sensors	Proximity Events	Device APIs		Likely to evolve substantially	Last updated September 2015   Commits on ed. draft	Very limited 		 Started

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Ambient Light sensor	Ambient Light Events	Device APIs		Likely to evolve significantly	Last updated September 2015  	Very limited 		 Started
Humidity sensor	Ambient Humidity Events	Device APIs	N/A	Unofficial draft	Last updated October 2013	None 		N/A
Generic Sensors	Generic Sensor API	Device APIs		Early draft	Last updated June 2015  	N/A		N/A
Camera & Microphone streams	Media Capture and Streams	Device APIs and Web Real-Time Communications		Stabilizing	Last updated August 2015  	Growing 		 started

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Bluetooth	Web Bluetooth	Web Bluetooth Community Group	Not on standards track	Early draft	Last updated August 2015   Commits on ed. draft	Experimental 		N/A

5. Network Integration

Interacting with the network is key to any Cloud-oriented application or service.

The Web platform is growing a number of APIs that facilitate establishing network connectivity in different contexts.

XMLHttpRequest (the basis for Ajax development) is a widely deployed API to load content from Web servers using the HTTP and HTTPs protocol: the W3C specification (formerly known as *XMLHttpRequest Level 2*) was meant to document the existing deployed API (with the ability to make requests on servers in a different domain, programmatic feedback on the progress of the network operations, and more efficient handling of binary content), but that work is now likely to be done only in the WHATWG. The WHATWG fetch API also provides a more powerful Promise-based alternative.

The ***Beacon*** API aims at letting developers queue unsupervised HTTP requests, leaving it to the browser to execute them when appropriate, opening the door for better network optimizations.

Early work on a ***Web Background Synchronization API*** would provide a robust Service Worker-based mechanism to enable Web applications to download and upload content in the background, even in the absence of a running browser.

By default, browsers do not allow to make request across different domains (or more specifically, across different *origins*, a combination of the protocol, domain and port) from a single Web page; this rule protects the user from having a Web site abusing their credentials and stealing their data on another Web site. Sites can opt-out of that rule by making use of the ***Cross-Origin Resource Sharing*** mechanism, opening up much wider cooperation across Web applications and services.

XMLHttpRequest is useful for client-initiated network requests, but mobile devices with their limited network capabilities and the cost that network requests induce on their battery (and sometimes on their users bill) can often make better use of server-initiated requests. The ***Server-Sent Events*** API allows triggering DOM events based on push notifications (via HTTP and other protocols.)

Early work on a **Push API** would allow Web applications to receive server-sent messages whether or not the said Web app is active in a browser window. An IETF Working Group charter is under discussion to standardize the protocol aspects of the mechanism.

The ***WebSocket API***, built on top of the IETF *WebSocket protocol*, offers a bidirectional, more flexible, and less resource intensive network connectivity than XMLHttpRequest.

The work on Web Real-Time Communications will also provide direct **peer-to-peer data connections** between browsers with real-time characteristics, opening the way to collaborative multi-devices Web applications.

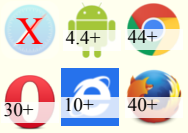
Of course, an important part of using network connectivity relies on being able to determine if such connectivity exists, and the type of network available. The HTML5 **onLine DOM flag** (and its associated change event, `ononline`) signals when network connectivity is available to the Web environment.

The network-information API, which was supposed to address discovery of the network characteristics, has been abandoned for the time being due to lack of clear supporting use cases.

The **Resource Timing API** offers to measure precisely the impact of the network on the time needed to load various resources, offering another approach to adapt a Web app to its network environment.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
HTTP(s) network API	XMLHttpRequest Level 1	Web Applications	WD	Likely to be abandoned in favor of WHATWG specification	Last updated May 2014	Well deployed 8+ 4.4.3+ 44+ 12+ 10+ 40+		Good coverage
	Web Background Synchronization	Web Applications	N/A	Early draft	Last updated April 2015  Commits on ed. draft	None X X X X X X		None
Cross-domain requests	Cross-Origin Resource Sharing	Web Applications and Web Application Security	Rec	Stable		Well-deployed 6.0+ 4.4+ 44+ 12+ 11+ 40+	WebPlatform.org	Well started
Server-pushed requests	Server-Sent Events	Web Applications	Rec	Stable	Finished	Getting well-deployed 4.0+ 4.4+ 44+ 11.1+ X 40+	WebPlatform.org	Good coverage
	Push API	Web Applications	WD	Early draft, now with Service Workers	Last updated August 2015  Commits on ed. draft	Limited X X 42+ X X X		N/A

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Bidirectional connections	The WebSocket API	Web Applications	CR	Stable	Last updated June 2014	Good deployment 6.0+ 4.4+ 44+ 12.1+ 10+ 40+	WebPlatform.org W3DEV CAMPUS	Good coverage
P2P data connections	WebRTC 1.0: Real-time Communication Between Browsers	Web Real-Time Communications	WD	Early draft	Last updated June 2015 O N D 2014 J F M A M J J A S 2015 Commits on ed. draft	Growing X 44+ 44+ 30+ X 40+	WebPlatform.org	Early start
on-line state	onLine state in HTML5	HTML	Rec	Stable	Finished	Limited X 2.2+ 18+ X 8+ X		Well started
Network characteristics	The Network Information API	Device APIs	Retired	Abandoned for now, but might be restarted	Last updated November 2014 O N D 2014 J F M A M J J A S 2015 Commits on ed. draft	Limited X 2.2+ 38+ X X 10+		None

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
	Resource Timing	Web Performance	WD	Stable	Last updated August 2015   Commits on ed. draft	Growing 		 Well started

6. Application Lifecycle

While Cloud services are potentially always in operation, their usage by end-users depend on their proper integration in the clients that they interact with, whose lifecycles depend on many parameters: battery, network connectivity, visibility on the device, etc.

These notions are part of the overall *application lifecycle*: how applications get installed, shown to the user in applications list, started, stopped, woken up from remote notifications, synced up when the device goes on-line.

These various capabilities are brought the Web platform through different mechanisms.

Although the notion of installed Web applications is still not well-defined, there are several components to the notion of installation that are under development.

Packaging on the Web describes a Web-adapted format to make Web content available in a singe file for ease of download, sharing or archiving.

Whether packaged or not, users rely on a variety of metadata (name, icons) to identify the apps they want to use among their list of regularly used applications. The JSON-based manifest format lets developers group all these metadata in a single JSON file.

HTML5's `ApplicationCache` enables access to Web applications off-line through the definition of a manifest of files that the browser is expected to keep in its cache.

While relatively well deployed, the current approach has shown some strong limitations in terms of how much developers can control what gets cached when. The Web Applications Working Group has thus been developing a more powerful approach, ServiceWorker.

Not only does Service Worker enables Web applications to work seamlessly off-line or in poor network conditions, it also creates a model for Web applications to operate when they have not been opened in a browser window, or even if the browser itself is not running.

That ability opens the door for Web applications that run in the background and can react to remotely triggered events.

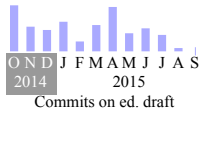
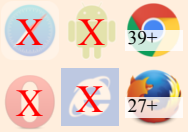
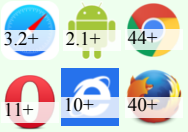
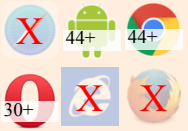
The Task Scheduler API makes it possible to trigger a task at a specified time via the Web app service worker. While the System Applications Working Group in which this API was developed has closed, the ServiceWorker-based approach taken in the specifications may make it an interesting starting point for further work in this space.

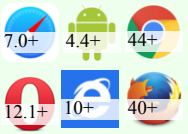
Similarly, the new geofencing API enables to wake up a Web app when a device enters a specified geographical area.

The Push API enables Web applications to subscribe to remote notifications that, upon reception, wake them up. Native applications have long enjoyed the benefits of greater user engagement that these notifications bring, and soon Web applications will share that ability.

Likewise, the Web Background Synchronization specification will enable Web applications to keep their user data up to date seamlessly, by running network operations in the background.

The Page Visibility specification lets developers detect when their application is in the foreground, and thus adapt their operations and resource consumption accordingly.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Packaging	Packaging on the Web	TAG and Web Applications	WD	Early draft	Last updated February 2015 	None 		N/A
	Manifest for a web application	Web Applications	WD	Early draft	Last updated August 2015 	Limited 		N/A
Offline Web Apps	ApplicationCache in HTML5	HTML	Rec	Stable (but Service Workers will be the preferred approach when available)	Finished	Well deployed 		 None
	Service Workers	Web Applications	WD	Early draft	Last updated September 2015 	Limited 		 Well started

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Scheduled tasks	Task Scheduler API Specification	System Applications	Retired	Early draft	Last updated October 2014  Commits on ed. draft	None 		None
Geofencing	Geofencing API	Geolocation		Just started	Last updated June 2015  Commits on ed. draft	None 		None
Remote Notifications	Push API	Web Applications		Early draft, now with Service Workers	Last updated August 2015  Commits on ed. draft	Limited 		N/A
Background Sync	Web Background Synchronization	Web Applications	N/A	Early draft	Last updated April 2015  Commits on ed. draft	None 		None
Foreground detection	Page Visibility	Web Performance		Finished		Well deployed 		Good coverage

7. Payment and Services

Our new W3C activity on payment is already looking at Cloud integration, eg. differences between eWallets that reside in your phone or in the cloud, or more generally any payment card details managed either on a secure element or on the cloud. Of course, the things people buy online, the actual data or resource may be outsourced to a cloud service provider and so communication and protocols must be developed in this context.

Meanwhile, HTML5.1 provides specific help for autocomplete of credit card details, making it easier to pay via credit cards once these details have been entered once.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Integrated payment	Credit card details autocomplete in HTML 5.1	HTML		Early draft	undefined	Very limited 		None

8. Performance & Tuning

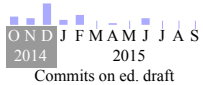
The work started by the Web Performance Working Group on ***Navigation Timing***, ***Resource Timing***, ***Performance Timeline*** and ***User Timing***, gives tools to Web developers for optimizing their Web applications. The work on the ***Frame Timing*** API aims at providing detailed information on the frame-per-second obtained when an application is running on the user device.

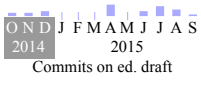
The *Resource Hints* and *Preload* specifications let developers optimize the download of resources by enabling to delay either the download or the execution of the downloaded resource.

The proposed work on Efficient Script Yielding offers the opportunity to Web developers to use more efficiently asynchronous programming, but has so far gained very limited traction.

The `requestIdleCallback` API similarly proposes a way for scheduling an operation at the next opportunity when the app is not processing another operation.

Beyond optimization of resources, the perceived reactivity of an application is also a critical aspect of the mobile user experience. The **thread-like mechanism** made possible via *Web Workers* allows keeping the user interface responsive by offloading the most resource-intensive operations into a background process.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Timing hooks	Navigation Timing	Web Performance	Rec	Finished		Well deployed		Good coverage
	Resource Timing	Web Performance	WD	Stable	Last updated August 2015  Commits on ed. draft	Growing		Well started
	Performance Timeline	Web Performance	Rec	Finished		Limited		Started
	User Timing	Web Performance	Rec	Finished		Growing		Well started
	Frame Timing	Web Performance	WD	Early draft	Last updated June 2015  Commits on ed. draft	None		None

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Network prioritization	Resource Hints	Web Performance	WD	Early draft	Last updated August 2015 	Growing deployment X 4+ 44+ 30+ 11+ 40+		None
	Preload	Web Performance	WD	Early draft	Last updated September 2015 	None?		None
Priority handling	Efficient Script Yielding	Web Performance	Editors	Early draft	Last updated April 2014	Very limited X X X X 10+ X		None
Threading	Web Workers	Web Applications	CR	Stable	Last updated May 2014	Well deployed 5.0+ 2.1+ 44+ 11+ 10+ 40+	WebPlatform.org W3DEV CAMPUS	Good coverage

9. Security & Privacy

Clearly a big intersection with the Cloud, and all Cloud programmers should follow this work if they want to write secure cloud web apps, concerned with identity, encryption, etc.

The first line of defense for users, and the unit of isolation for Web apps is the same-origin policy that roughly limits what a Web application can access to content and data hosted on the same origin, i.e. the combination of URL scheme, domain name and port.

For legacy reasons, this policy is not as stringent on some parts of the Web platform, exposing users to greater attack surface via cross-site scripting or cross-site request forgery. To enable Web application authors to reduce the attack surface beyond what legacy requires, the ***Content Security Policy (level 2)*** offers hooks that severely limits damages that an attacker could hope to achieve.

To further strengthen the integrity of their applications, Web developers can make use of the proposed ***Subresource integrity*** mechanism, that makes it possible to block man-in-the-middle attacks or compromised third-parties providers.

Entry Point Regulation provides another layer of strengthening and offers to filter the type of HTTP requests that can be made from external sites, reducing risks of cross-site script and cross-site request forgery.

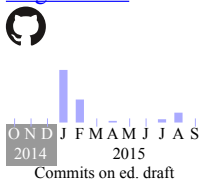
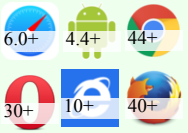
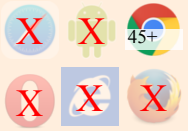
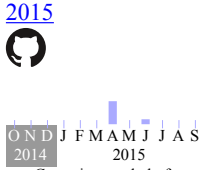
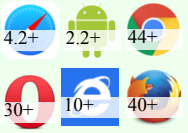
In applications that aggregate content from multiple (possibly untrusted) sources, the HTML5 `iframe` sandbox makes it possible to restrict what kind of interactions third-party embedded content can make use of.

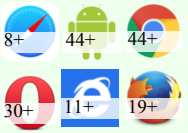
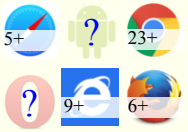
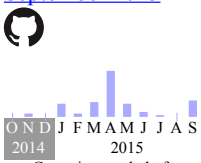
As described earlier, the Web Cryptography API provides the necessary tools to encrypt data for storage and transmission from within Web applications, with access pre-provisioned keys via the WebCrypto Key Discovery API.

There are discussions to bring the capabilities of hardware-security modules to the Web, to enable access to high-security operations for encryption, payment, identity proof, etc., embodied in a draft charter for a Hardware Security Working Group.

For users that wish to indicate their preferences not to be tracked across Web applications and sites, the Tracking Preference Expression (also known as Do No Track) enables browsers to communicate explicitly their wish to content providers, and to determine whether a given content provider asserts fulfilling that wish.

To facilitate the authentication of users to on-line services, the Web Application Security Working Group is proposing a credential management API that lets developers interact more seamless with user-agent-managed credentials.

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Strengthened security	Content Security Policy Level 2	Web Application Security	CR	Stable	Last updated August 2015 	Well-deployed 		 Well started
	Subresource Integrity	Web Application Security	WD	Just started	Last updated August 2015 	Limited 		None
	Entry Point Regulation	Web Application Security	WD	Just started	Last updated June 2015 	None 		None
	Sandboxed iframe in HTML5	HTML	Rec	Stable	Finished	Widely deployed 		 None

Feature	Specification	Working Group	Maturity	Stability	Latest editors draft	Current implementations	Developers doc	Test suite
Encryption	Web Cryptography API	Web Cryptography	CR	Stable	Last updated November 2014 	Well deployed 		 Early start
	WebCrypto Key Discovery	Web Cryptography	WD	Early work	Last updated May 2014	None 		None
Tracking protection	Tracking Preference Expression (DNT)	Tracking Protection	CR	Stabilizing	undefined	Good deployment 		None
Identity management	Credential Management Level 1	Web Application Security	WD	Early draft	Last updated September 2015 	None 		N/A

Acknowledgments

Thanks to Art Barstow, Anssi Kostiainen, Jo Rabin, J. Manrique López, Mounir Lamouri, Marcos Caceres, François Daoust and Ronan Cremin for their contributions to this document.

This document is produced through the HTML5Apps project, funded by the European Union through the Seventh Framework Programme (FP7/2013-2015) under grant agreement n°611327 - HTML5 Apps.