WEB APIS YOU probably DIDN'T KNOW EXISTED

@zenorocha
HTML 5
A vocabulary and associated APIs for HTML and XHTML

W3C Working Draft 22 January 2008

This Version: http://www.w3.org/TR/2008/WD-html5-20080122/
Latest Published Version: http://www.w3.org/TR/html5/
Editors: Ian Hickson, Google, Inc.
David Hyatt, Apple, Inc.

Copyright © 2008 W3C® (MIT, ERCIM, Keio). All Rights Reserved. W3C liability, trademark and document use rules apply.

Abstract

This specification defines the 5th major revision of the core language of the World Wide Web, HTML. In this version, new features are introduced to help Web application authors, new elements are introduced based on research into prevailing authoring practices, and special attention has been given to defining clear conformance criteria for user agents.

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/.

If you wish to make comments regarding this document, please send them to public-html-comments@w3.org (subscribe, archives). All feedback is welcome.

Implementors should be aware that this specification is not stable. Implementors who are not taking part in the discussions are likely to find the specification changing out from under them. At this point, these changes are not yet part of a stable release.

If you’re a web developer looking forward to the new tools in HTML 5, the next generation of the language that powers the web, we have some bad news for you — you’re going to waiting a while.

Ian Hickson, the editor of the HTML 5 specification, recently outlined the time table for HTML 5 and, even assuming browser manufacturers embrace HTML 5 when it reaches the final draft stage, that puts HTML 5’s widespread adoption at 2012. Worse, the final proposed...
2010
Thoughts on Flash

Apple has a long relationship with Adobe. In fact, we met Adobe’s founders when they were in their proverbial garage. Apple was their first big customer, adopting their Postscript language for our new Laserwriter printer. Apple invested in Adobe and owned around 20% of the company for many years. The two companies worked closely together to pioneer desktop publishing and there were many good times. Since that golden era, the companies have grown apart. Apple went through its near death experience, and Adobe was drawn to the corporate market with their Acrobat products. Today the two companies still work together to serve their joint creative customers – Mac users buy around half of Adobe’s Creative Suite products – but beyond that there are few joint interests.

I wanted to jot down some of our thoughts on Adobe’s Flash products so that customers and critics may better understand why we do not allow Flash on iPhones, iPods and iPads. Adobe has characterized our decision as being primarily business driven – they say we want to protect our App Store – but in reality it is based on technology issues. Adobe claims that we are a closed system, and that Flash is open, but in fact the opposite is true. Let me explain.

First, there’s “Open”.

Adobe’s Flash products are 100% proprietary. They are only available from Adobe, and Adobe has sole authority as to their future enhancement, pricing, etc. While Adobe’s Flash products are widely available, this does not mean they are open, since they are controlled entirely by Adobe and available only from Adobe. By almost any definition, Flash is a closed system.

Apple has many proprietary products too. Though the operating system for the iPhone, iPod and iPad is proprietary, we strongly believe that all standards pertaining to the web should be open. Rather than use Flash, Apple has adopted HTML5, CSS and JavaScript – all open standards. Apple’s mobile devices all ship with high performance, low power implementations of these open standards. HTML5, the new web standard that has been adopted by Apple, Google and many others, lets web developers create advanced graphics, typography, animations and transitions without relying on third party browser plug-ins (like Flash). HTML5 is completely open and controlled by a standards committee, of which Apple is a member.
Mark Zuckerberg: Our Biggest Mistake Was Betting Too Much On HTML5

Posted Sep 11, 2012 by Drew Olanoff (@drew)

Today, Mark Zuckerberg revealed that Facebook's mobile strategy relied too much on HTML5, rather than native applications.

Not only was this a big mistake with mobile, but Zuckerberg says that its biggest mistake period was the focus on HTML5. This is the first time that the Facebook CEO has openly admitted this, but things are looking good for the new iOS native app.

According to Zuckerberg, people are consuming twice as many feed stories since the update to the new iOS app, which is great.

*The first half year has been a little bit slow on product, but for the next six months I expect a lot of really cool stuff.*

This "really cool stuff" will probably have monetization in mind, as it's very clear that mobile is the path to ad revenue for the company.

It's extremely difficult for a company to nose-dive into an adoption of a particular set of tools and then quickly change course. I suspect that this is exactly what happened with Facebook and things are at least looking up.
It’s been a fantastic couple of years for HTML5 games, for both consumers and developers. In fact, HTML5 is turning into a great game development platform, rapidly catching up to the ubiquity of Flash-based browser gaming.

HTML5 comes with some huge advantages that allow users to build apps, accelerated graphics in games, stream HD video and so much more by just using native web code.

So, here’s a collection of the 30 most creative and addictive HTML5 games, all of which can be played in your web browser. But, beware, they are extremely addictive!

1) SAND TRAP

Fill the bucket with sand. Sounds simple enough but it’s actually trickier than you think as there’s a rotating cube maze that hovers above the bucket.
2014
Open Web Platform Milestone Achieved with HTML5 Recommendation

Next Generation Web Technologies Build on Stable Foundation

28 October 2014 — The World Wide Web Consortium (W3C) published a Recommendation of HTML5, the fifth major revision of the format used to build Web pages and applications, and the cornerstone of the Open Web Platform. For application developers and industry, HTML5 represents a set of features that people will be able to rely on for years to come. HTML5 is now supported on a wide variety of devices, lowering the cost of creating rich applications to reach users everywhere.

"Today we think nothing of watching video and audio natively in the browser, and nothing of running a browser on a phone," said Tim Berners-Lee, W3C Director. "We expect to be able to share photos, shop, read the news, and look up information anywhere, on any device. Though they remain invisible to most users, HTML5 and the Open Web Platform are driving these growing user expectations."

HTML5 brings to the Web video and audio tracks without needing plugins; programmatic access to a resolution-dependent bitmap canvas, which is useful for rendering graphs, game graphics, or other visual images on the fly; native support for scalable vector graphics (SVG) and math (MathML); annotations important for East Asian typography (Ruby); features to enable accessibility of rich applications; and much more.

HTML5 is Widely Deployed

HTML5 has been in use for years. According to a 2014 Vision Mobile Survey, 42% of 10,000 developers surveyed are using the combination of HTML, CSS, and JavaScript for all or part of their mobile applications. Gartner identified HTML5 as one of their top 10 mobile technologies and capabilities for 2015 and 2016, saying HTML5 "will be an essential technology for organizations delivering applications across multiple platforms."

To help achieve the "write once, deploy anywhere" promise of HTML5 and the Open Web Platform, during the 22 months since W3C announced the completed definition of HTML5, the W3C community has been adding to the HTML5 test suite, which includes over 100,000 tests and continues to grow. The Test the Web Forward community effort now plays an important and ongoing part in driving Open Web Platform interoperability.

With today's publication of the Recommendation, software implementers benefit from Royalty-Free licensing commitments from over sixty companies under W3C’s Patent Policy. Enabling implementers to use Web technology without payment of royalties is critical to making the Web a platform for innovation.

What’s Next: Application Foundations for Developers, New Use Cases for the Web
Now what?
SVG  CANVAS  WEB  LOCAL  STORAGE  VIDEO  SOCKETS  AUDIO  WEBRTC  DRAG & DROP  GEOLOCATION
PAGE
visibility
Provides an API to ask whether the current page is visible or not.
window.addEventListener('visibilitychange', () => {
  if (document.hidden) {
    console.log('Tab is hidden');
  }
  else {
    console.log('Tab is focused');
  }
});
Page Visibility 2
W3C First Public Working Draft 23 June 2016

This version:
http://www.w3.org/TR/2016/WD-page-visibility-2-20160623/

Latest published version:
http://www.w3.org/TR/page-visibility-2/

Latest editor’s draft:
https://w3c.github.io/page-visibility/

Editors:
Ilya Grigorik, Google, igrigorik@gmail.com
Arvind Jain, Google Inc., (Until December 2014)
Jalinder Mann, Microsoft Corp. (Until February 2014)

Repository:
We are on GitHub.
File a bug.
Commit history.

Mailing list:
public-web-perf@w3.org

Implementation:
Can I use Page Visibility?
Test Suite
Test Suite repository

Copyright © 2016 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and document use rules apply.

Abstract

This specification defines a means to programmatically determine the visibility state of a document. This can aid
with tasks such as preventing overlaps of documents, and improving performance.
window.addEventListener('visibilitychange', () => {
    switch(document.visibilityState) {
        case 'prerender':
            console.log('Tab is pre-rendering');
            break;
        case 'hidden':
            console.log('Tab is hidden');
            break;
        case 'visible':
            console.log('Tab is focused');
            break;
    }
});
WHERE
to use it?
ONLINE STATE

Exposes a network connection availability information to the web.
ONLINE STATE

close.log(navigator.onLine ? 'online' : 'offline')
window.addEventListener('offline', networkStatus);
window.addEventListener('online', networkStatus);

function networkStatus(e) {
    console.log(e.type);
}
BROWSER support

caniuse.com/#feat=online-status
Network Information API
Living Document

W3C Editor's Draft 24 May 2016

This version:
https://w3c.github.io/netinfo/

Latest published version:
https://www.w3.org/TR/netinfo-api/

Latest editor's draft:
https://w3c.github.io/netinfo/

Editors:
Marcos Cáceres, Mozilla Corporation
Fernando Jiménez Moreno, Telefonica
Ilya Grigorik, Google

Repository:
We are on Github.
File a bug.
Commit history.

Implementations:
Chromium

Copyright © 2016 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and document use rules apply.

Abstract
The Network Information API enables web applications to access information about the network connection in use by the device.
WHERE to use it?
VIBRATION
VIBRATION

Provides access to a form of tactile feedback.
// Vibrate for 1 second
navigator.vibrate(1000);

// Vibrate with a pattern
navigator.vibrate([400, 300, 300, 200, 500]);

// Turn off vibration
navigator.vibrate(0);
// Super Mario
navigator.vibrate([[125, 75, 125, 275, 200, 275, 125, 75, 125, 275, 200, 600, 200, 600]]);

// Star Wars
navigator.vibrate([[500, 110, 500, 110, 450, 110, 200, 110, 170, 40, 450, 110, 200, 110, 170, 40, 500]]);

// Go Go Power Rangers
navigator.vibrate([[150, 150, 150, 150, 75, 75, 150, 150, 150, 150, 150, 150, 150, 150, 450]]);

https://goo.gl/bX4ZQv
WHERE

to use it?
DEVICE orientation
DEVICE ORIENTATION

Provides access to device's physical orientation.
window.addEventListener('deviceorientation', (e) => {
  console.log('Gamma:', e.gamma);
  console.log('Beta:', e.beta);
  console.log('Alpha:', e.alpha);
});
let logo = document.querySelector('img');

window.addEventListener('deviceorientation', (e) => {
  let tiltLR = e.gamma;
  let tiltFB = e.beta;

  logo.style.transform = `rotate(${tiltLR}deg)
                          rotate3d(1,0,0, ${tiltFB * -1}deg)`;
});
BROWSER support

caniuse.com/#feat=deviceorientation
Não tem quem duvide que Minas Gerais é uma terra de grandes belezas. Terra de montanhas que acolhem, de história que ensina e de um povo
clipboard

copy & paste
Clipboard

Standard APIs for interacting with the clipboard (copy/cut/paste).
ZeroClipboard v2.x

The ZeroClipboard library provides an easy way to copy text to the clipboard using an invisible Adobe Flash movie and a JavaScript interface.

The "Zero" signifies that the library is invisible and the user interface is left entirely up to you.

Looking for v1.x? Testing this page locally
// 1. User interaction is required

let button = document.querySelector('button');

button.addEventListener('click', () => {
    select();
    copy();
});
// 2. Programatically select an element

function select() {
    let input = document.querySelector('input');

    input.focus();
    input.setSelectionRange(0, input.value.length);
}

// CLIPBOARD
// 3. Copy selected element text

function copy() {
    try {
        document.execCommand('copy');
    } catch (err) {
        console.error(err);
    }
}
```
// Clipboard event listeners

document.addEventListener('copy', (e) => {
  console.log(e.target.value);
});

document.addEventListener('cut', (e) => {
  console.log(e.target.value);
});

document.addEventListener('paste', (e) => {
  console.log(e.clipboardData.getData('text/plain'));
});
```
clipboard.js
A modern approach to copy text to clipboard
No Flash. No frameworks. Just 3kb gzipped

Why
Copying text to the clipboard shouldn't be hard. It shouldn't require dozens of steps to configure or hundreds of KBs to load.
But most of all, it shouldn't depend on Flash or any bloated
WHERE to use it?
Modern copy to clipboard. No Flash. Just 3kb gzipped [https://clipboardjs.com](https://clipboardjs.com)

<table>
<thead>
<tr>
<th>Branch</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>master</td>
<td>Adds basic JSFiddle to help people reproduce their bugs</td>
<td>3 months ago</td>
</tr>
<tr>
<td>demo</td>
<td>Updates delegate package which now exposes e.delegateTarget property</td>
<td>9 months ago</td>
</tr>
<tr>
<td>dist</td>
<td>Release v1.8.12</td>
<td>2 months ago</td>
</tr>
<tr>
<td>src</td>
<td>Fix bug that unable to remove fake event listener (#258)</td>
<td>2 months ago</td>
</tr>
<tr>
<td>test</td>
<td>Removes error message when target/text attributes are null, undefined...</td>
<td>4 months ago</td>
</tr>
<tr>
<td>babelrc</td>
<td>update modules and fix #176</td>
<td>5 months ago</td>
</tr>
<tr>
<td>.banner</td>
<td>Adds banner - Fixes #88</td>
<td>10 months ago</td>
</tr>
<tr>
<td>.editorconfig</td>
<td>Add .npmignore</td>
<td>10 months ago</td>
</tr>
<tr>
<td>.gitignore</td>
<td>Provides a transpiled version to npm so browserify and webpack users ...</td>
<td>9 months ago</td>
</tr>
<tr>
<td>.npmignore</td>
<td>Ignores unnecessary files on bower and npm</td>
<td>9 months ago</td>
</tr>
</tbody>
</table>
Components
Over a dozen reusable components built to provide buttons, dropdowns, input groups, navigation, alerts, and much more.

Pagination
Provide pagination links for your site or app with the multi-page pagination component.

Contents
- Overview
- Disabled and active states
- Sizing

Overview
ambient
LIGHT
AMBIENT LIGHT

Exposes sensor data that captures the light intensity.
window.addEventListener('devicelight', (e) => {
  console.log(`<${e.value} lux`);
});
let sensor = new AmbientLightSensor();
sensor.start();

sensor.onchange = (e) => {
  console.log(e.reading.illuminance);
};
sensor.stop();
BROWSER support
caniuse.com/#feat=ambient-light
WHERE to use it?
battery STATUS
BATTERY STATUS

Allows a web page to access battery information from desktop and mobile devices.
navigator.getBattery().then((battery) => {
  console.log(`\${battery.level * 100}%`);
  battery.addEventListener('levelchange', () => {
    console.log(`\${this.level * 100}%`);
  });
});
BROWSER support

caniuse.com/#feat=battery-status
WHERE to use it?
Catch Omanyte!
You got 23min remaining until the battery dies.
WEB COMPONENTS

templates  shadow dom  html imports
custom elements
progressive web apps

service workers

background sync

push notifications

app manifest

offline support
WEB
assembly
WebAssembly, or wasm, is a low-level programming language for the client-side.
WEBVR
Experimental API that provides access to Virtual Reality devices, such as the Oculus Rift or Google Cardboard.
BROWSER support

[Icons of various browsers]

chromestatus.com/features#webvr
Gives access to a game controller via USB.
window.addEventListener('gamepadconnected', () => {
    let gp = navigator.getGamepads()[0];

    console.log('ID:', gp.id);
    console.log('Axes:', gp.axes.length);
    console.log('Buttons:', gp.buttons.length);
});
window.addEventListener('gamepadconnected', gameLoop);

function gameLoop() {
    let gp = navigator.getGamepads()[0];

    if (gp.buttons[0].pressed) {
        console.log('X');
    }

    requestAnimationFrame(gameLoop);
}

**Gamepad API additions: Touched and Pose**

**Change description:**
Add a couple of properties to the Gamepad API: a "touched" property for GamepadButton to explicitly support touchpads and a "pose" object on the Gamepad itself to support devices with accelerometers and 3 or 6 degree of freedom tracking. Our immediate interest in these new properties is to support VR controllers such as the Daydream controller, Vive wands, and Oculus Touch, but they are also applicable to a wide variety of more traditional gamepads.

**Changes to API surface:**
* "touched" boolean added to GamepadButton
* "pose" object added to Gamepad
  * includes 6 Float32Arrays, each representing a vector or quaternion: position, orientation, linearAcceleration, linearVelocity, angularAcceleration, angularVelocity.

**Links:**
- Public standards discussion:
  - [https://github.com/w3c/gamepad/pull/26](https://github.com/w3c/gamepad/pull/26)
  - [https://github.com/w3c/gamepad/pull/25](https://github.com/w3c/gamepad/pull/25)

**Support in other browsers:**
- **Internet Explorer:** No
- **Firefox:** No
- **Safari:** No

*Make sure to fill in any labels with a -?, including all OSes this change affects. Feel free to leave other labels at the defaults.*
BROWSER support

caniuse.com/#feat=gamepad
INSERT A CARTRIDGE TO START PLAYING
THANK YOU
so much

@zenorocha