Review of the HTML5 API

Computer Systems Department, University of Castilla-La Mancha
Albacete, Spain
felix.albertos@uclm.es

5th June 2018
Index

Introduction

API

Work in Progress

Hands on

Conclusions
Index

- Introduction
- API
- Evolution
- Hands on
- Conclusions
Review of the HTML5 API

Félix Albertos Marco

ICWE 2018 TUTORIAL
Review of the HTML5 API
Review of the HTML5 API
Semantics

- Giving meaning to structure
Offline & Storage

- App Cache
- Local Storage
- Session Storage
- Indexed DB
- File API
- Cache Storage
Device Access

- Geolocation API
- Audio/Video input
- Contacts/Events
- Tilt Orientation
Connectivity & Communication

- Server Sent Events
- Web Sockets
- WebRTC
- Cross-document messaging
- Channel messaging
- Broadcasting to other browsing contexts
Multimedia

- Audio
- Video
3D Graphics, Effects

- SVG
- Canvas
- WebGL
- CSS3 3D
- Frame Animation
Performance & Integration

- Web Workers
- XMLHttpRequest 2
- Service Workers
Web Page

Web Application

Native Web Application
Index

- Introduction
- API
- Work in Progress
- Hands on
- Conclusions
Method that enables applications to take advantage of persistent background processing, including hooks to enable bootstrapping of web applications while offline.
Design Principles

- Compatibility
- Utility
- Interoperability
- Universal Access

HTML Design Principles
W3C Working Draft 26 November 2007

This Version:
http://www.w3.org/TR/2007/WD-html-design-principles-20071126/
Design Principles

Compatibility Utility Interoperability Universal Access

• Support Existing Content

• Is there a significant quantity of existent content on popular websites intended for consumption on the public web currently working as intended in multiple popular user agents?
Design Principles

Compatibility Utility Interoperability Universal Access

• Degrade Gracefully

• Web content can degrade gracefully in older or less capable user agents, even when making use of new elements, attributes, APIs and content models.
• Web browsers:
  • Current top mainstream
  • Highly popular old Web browsers
  • Top user agents designed to meet specific needs or address specialized markets
Design Principles

**Compatibility** Utility Interoperability Universal Access

- Do not Reinvent the Wheel
- Pave the Cowpaths
- Evolution Not Revolution
Design Principles

Compatibility Utility Interoperability Universal Access

• Solve Real Problems

• Priority of Constituencies
  • Users > authors > implementors > specifiers > theoretical purity

• Secure By Design
  • Communicating between documents from different sites is useful, but an unrestricted version could put user data at risk.
Design Principles

Compatibility Utility Interoperability Universal Access

• Separation of Concerns
  • Content <-- Presentation
  • article element do not defines details of how it is displayed
  • b and I elements are widely used

• DOM Consistency
Design Principles

Compatibility Utility **Interoperability** Universal Access

- Well-defined Behaviour
- Avoid Needless Complexity
- Handle Errors
Design Principles

Compatibility Utility Interoperability **Universal Access**

- Media Independence
  - work across different platforms, devices and media
- Support World Languages
- Accessibility
  - Access by everyone regardless of ability is essential
HTML 5 first working draft released

W3C Publishes HTML 5 Draft, Future of Web Content

Web Community Forges Next HTML Standard in Public W3C Forum

Contact Americas, Australia --
Ian Jacobs, <ij@w3.org>, +1.718.260.9447 or +1.617.253.2613

Contact Europe, Africa and the Middle East --
Marie-Claire Forgue, <mcf@w3.org>, +33.492.38.75.94

Contact Asia --
Yasuyuki Hirakawa <chibao@w3.org>, +81.466.49.1170

(Also available in French and Japanese; see also translations in other languages)

http://www.w3.org/ -- 22 January 2008 -- W3C today published an early draft of HTML 5, a major revision of the markup language for the Web. The HTML Working Group is creating HTML 5 to be the open, royalty-free specification for rich Web content and Web applications. The group operates entirely in public with nearly five hundred participants, including representatives from W3C Members ACCESS, AOL, Apple, Google, IBM, Microsoft, Mozilla, Nokia, and Opera.

"HTML is of course a very important standard," said Tim Berners-Lee, author of the first version of HTML and W3C Director. "I am glad to see that the community of developers, including browser vendors, is working together to create the best possible path for the Web. To integrate the input of so many people is hard work, as is the challenge of balancing stability with innovation, pragmatism with idealism."
The standard is in continuous development

- Added elements
- Removed elements
- Changed elements

- **Obsolete**
  This feature is obsolete. Although it may still work in some browsers, its use is discouraged since it could be removed at any time. Try to avoid using it.

- **This feature has been removed. Please fix your Web sites and applications.**
  This method was removed in Chrome 43, and Firefox 56.
Timeline

- Dec 1999: HTML 4.01
- Nov 2007: HTML Design Principles
- Jan 2008: HTML5 first draft
- Oct 2014: HTML5
- Oct 2015: HTML5.1
- Oct 2017: HTML5.2
HTML5

A vocabulary and associated APIs for HTML and XHTML

W3C Recommendation 28 October 2014

This Version:
http://www.w3.org/TR/2014/REC-html5-20141028/
Timeline

- Dec 1999: HTML 4.01
- Nov 2007: HTML Design Principles
- Jan 2008: HTML5 first draft
- Oct 2014: HTML5
- Oct 2017: HTML5.1
- Dec 2017: HTML5.2
HTML 5.1

W3C Recommendation 3 October 2017 (2nd Edition)

HTML 5.1 2nd Edition
W3C Recommendation 3 October 2017

This version:
https://www.w3.org/TR/2017/REC-html51-20171003/
HTML 5.1

Features added (some ...)

- picture and srcset attributes allow responsive image selection
- menuitem and type="context" attribute value enable authors to add functionality to the browser’s context menu
- requestAnimationFrame API allows for more efficient animation
- enqueueJob and nextJob help explain Promise resolution in terms of microtasks
- HTMLMediaElement and srcObject objects
- event-source-error, event-track-error and event-track-load events for media fetching
- history.scrollRestoration to control where a users' view is directed when navigating through their history
- add "noopener" to rel and window to allow for browsing contexts to be separated
- nonce attribute on script and style to support the use of Content Security Policy
HTML 5.1

Features added (some ...)

picture and srcset attributes allow responsive image selection

• Srcset & sizes
  • Identical images
  • Larger or smaller depending on the device
  • For saving bandwith
• To test:
  • Use incognito mode in chrome
  • Load with selected size and resize using mobile view
  • From the smallest width, increase width and notice image resolution changes
  • When page with more quality has been loaded, it always show it because it is intended for saving data
HTML 5.1

Features added (some …)

picture and srcset attributes allow responsive image selection
HTML 5.1

Features added (some ...)

`picture` and `srcset` attributes allow responsive image selection

```html
<picture>
  <source media="(max-width: 80px)" srcset="img/WinnerPodiumClose.jpg">
  <source media="(min-width: 80px)" srcset="img/WinnerPodium.jpg">
  <img src="img/WinnerPodium.jpg">
</picture>
```
HTML 5.1

Features added (some ...)

requestAnimationFrame API allows for more efficient animation
- Execution of code on the **next available screen repaint**
- Usually 60 times per second
  - Generally match the display refresh rate in most web browsers as per W3C recommendation
- `window.requestAnimationFrame(callback)`
  - callback takes one single argument which indicates the current time (in milliseconds)
HTML 5.1

Features added (some ...)

requestAnimationFrame API allows for more efficient animation
HTML 5.1

Features added (some ...)

`history.scrollRestoration` to control where a user's view is directed when navigating through their history

- `history.scrollRestoration = 'auto'`;
  - The browser restores the previous scroll position
- `history.scrollRestoration = 'manual'`;
  - The scroll is not restored by the browser
HTML 5.1

Features added (some ...)

add "noopener" to rel and window to allow for browsing contexts to be separated

<p>Pick the yellow peach that looks like a sunset with its red, orange, and pink coat skin, peel it off with your teeth. Sink them into unripened... Please, <a href="../magazine/index.html" rel="noopener" target="_blank">visit my site</a> for more information :)</p>
HTML 5.1

Features added (some ...)

add "noopen" to rel and window to allow for browsing contexts to be separated

```html
<p>Pick the yellow peach that looks like a sunset with its red, orange, and pink coat skin, peel it off with your teeth. Sink them into unripened... Please, <a href="../magazine/index.html" target="_blank">visit my site</a> for more information :)</p>
```
HTML 5.1

Features added (some ...)

add "noopener" to rel and window to allow for browsing contexts to be separated
HTML 5.1

Features added (some …)

add "noopener" to rel and window to allow for browsing contexts to be separated
HTML 5.1

Features added (some ...)

add "noopener" to rel and window to allow for browsing contexts to be separated
HTML 5.1

Features added (some ...)

nonce attribute on script and style to support the use of Content Security Policy

- What is Content Security Policy?
  - Security layer to detect and mitigate attacks
    - Cross Site Scripting (XSS)
    - Data Injection Attacks
HTML 5.1

Features added (some ...)

nonce attribute on script and style to support the use of Content Security Policy

• How to enable it?
  • HTTP header
    • Content-Security-Policy: policy
  • or <meta> element
    • <meta http-equiv="Content-Security-Policy" content="default-src 'self'; img-src https:///; child-src 'none';">
Features added (some ...)

nonce attribute on script and style to support the use of Content Security Policy

- script-src policy to allow sources
  - Content-Security-Policy: script-src <source> <source>;

- For example:
  - CSP header:
  - Script
    - <script src="http://www.felixalbertos.com/script/foobar/js/boot.js"></script>
HTML 5.1

Features added (some ...)

nonce attribute on script and style to support the use of Content Security Policy

- Among other values, source could be
  - nonce-<base64-value>
  - A whitelist for specific inline scripts using a cryptographic once (number used once)
    - Dynamically generated, for each requested page, if it is static ...
    - For example:
      - Content-Security-Policy: script-src ‘nonce-kjl93204ids’
      - <script nonce=“kjl93204ids”>
        // JavaScript code
      </script>
HTML 5.1

Features removed (some ...)

- appCache
  - Offline Web Applications, almost ...
- Media Controllers
- command API
  - A command is the abstraction behind menu items, buttons, and links. Once a command is defined, other parts of the interface can refer to the same command, allowing many access points to a single feature
- usemap attribute on object

Why?
- Sometimes nobody, or only one (Media Controllers !), has implemented it
- Implementation is inconsistent
- It doesn’t work as expected
HTML 5.1

Features removed (some ...)

• appCache
  • Not able to decide if serving cached or remote content
  • Not having control of which files are being processed
  • If something fails, it totally breaks
HTML 5.1

Features changed (some ...)

• accesskey takes a single character as a value (as in HTML 4)
  • accesibility
• header and footer elements can be nested, if each level is within a sectioning element
• mousewheel event is called wheel
• img and related elements support width="0"
• .tFoot and .createTFoot() always insert at the end of a table
HTML 5.2

W3C Recommendation 14 December 2017

This version:

https://www.w3.org/TR/2017/REC-html52-20171214/
HTML 5.2

Features added (some ...)

• `<dialog>` element

• Integration with the JavaScript module system of [ECMA-262]

• Update ARIA reference to [wai-aria-1.1], which introduces new features to improve accessibility

• `allowpaymentrequest` attribute of `iframe`, for integration with the Payment Request API [PAYMENT-REQUEST]

• `allow-presentation` value for the `sandbox` attribute of `iframe`, for integration with the Presentation API [PRESENTATION-API]

• Define the about:html-kind URL for MP4 media track integration
HTML 5.2

Features added (some ...)

<dialog> element

- The dialog element represents a part of an application that a user interacts with to perform a task, for example a dialog box, inspector, or window

- The open attribute indicates that the dialog is active and available for interaction. When the open attribute is not set, it shouldn't be shown to the user
  - The method show() / showModal() opens the dialog

- Closing the dialog
  - method close([return_value])
  - Property returnValue of the dialog receives the value in return_value

- When it is used with integrated forms, the submit button closes the dialog
HTML 5.2

Features added (some …)

<dialog> element

```javascript
function openDialog(id, modal){
    var userID = id;
    var userNameDialog = document.getElementById("userNameDialog");
    userNameDialog.innerHTML = "";
    if (modal){
        myDialog.showModal();
    }else{
        myDialog.show();
    }
}
```

```javascript
function closeDialog(v){
    if (v){
        document.getElementById("password").value = pwd;
        var elem = document.getElementById(userID);
        elem.parentNode.removeChild(elem);
        myDialog.close();
    }else{
        myDialog.close();
    }
}
```

```html
<button onclick="openDialog('user-1',true);">Delete account</button>
```
HTML 5.2

Features added (some ...)

- Integration with the JavaScript module system of [ECMA-262]

JavaScript

Some parts of the language described by this specification only support JavaScript as the underlying scripting language. [ECMA-262]

NOTE:
The term "JavaScript" is used to refer to ECMA262, rather than the official term ECMAScript, since the term JavaScript is more widely known. Similarly, the MIME type used to refer to JavaScript in this specification is text/javascript, since that is the most commonly used type, despite it being an officially obsoleted type according to RFC 4329. [RFC4329]
**HTML 5.2**

Features added (some ...)

- **allowpaymentrequest attribute of iframe, for integration with the Payment Request API** [PAYMENT-REQUEST]
  - To indicate that a cross-origin iframe is allowed to invoke the payment request API, the allowpaymentrequest attribute can be specified on the iframe element.

- **Payment-Request**
  - This specification standardizes an API to allow merchants (i.e. web sites selling physical or digital goods) to utilize one or more payment methods with minimal integration. User agents (e.g., browsers) facilitate the payment flow between merchant and user.
HTML 5.2

Features added (some ...)

- allowpaymentrequest attribute of iframe, for integration with the Payment Request API [PAYMENT-REQUEST]

- Payment-Request
  - Fast purchase experience
  - Consistent experience on every site (that supports the API)
  - Accessibility
  - Credentials management
  - Consistent error handling
HTML 5.2

Features added (some ...)

- allowpaymentrequest attribute of iframe, for integration with the Payment Request API [PAYMENT-REQUEST]
- Payment-Request

Source: https://developers.google.com/web/fundamentals/payments/deep-dive-into-payment-request
HTML 5.2

Features added (some …)

• allow-presentation value for the sandbox attribute of iframe, for integration with the Presentation API [PRESENTATION-API]
  • Allows embedders to have control over whether an iframe can start a presentation session

• Presentation API
  • Enable Web content to access presentation displays and use them for presenting Web content
  • Supported types of multimedia devices include both displays which are wired using HDMI, DVI, or the like, or wireless, using DLNA, Chromecast, AirPlay, or Miracast.
Features added (some ...)

- allow-presentation value for the sandbox attribute of iframe, for integration with the Presentation API
  
  [PRESENTATION-API]
- Presentation API
### HTML 5.2

**Features added (some ...)**

- allow-presentation value for the sandbox attribute of iframe, for integration with the Presentation API ([PRESENTATION-API](https://googlechrome.github.io/samples/presentation-api/index.html))
- Presentation API

```javascript
const presentationRequest = new PresentationRequest(['receiver/index.html']);
// Make this presentation the default one when using the "Cast" browser menu,
navigator.presentation.defaultRequest = presentationRequest;

document.querySelector('#start').addEventListener('click', function() {
  log('Starting presentation request...');
presentationRequest.start();
});
```

Source: https://googlechrome.github.io/samples/presentation-api/index.html
HTML 5.2

Features added (some ...)

- allow-presentation value for the sandbox attribute of iframe, for integration with the Presentation API [PRESENTATION-API]
- Presentation API

```javascript
document.querySelector('#sendMessage').addEventListener('click', function() {
    const message = document.querySelector('#message').value.trim();
    const lang = document.body.lang || 'en-US';
    log('sending "" + message + "...";
    presentationConnection.send(JSON.stringify({message, lang}));
});
```

```javascript
connection.addEventListener('message', function(event) {
    const data = JSON.parse(event.data);
    const logString = 'Message ' + messageIdx + ' from connection ' + connection.connectionId + ': ' + data.message;
    addMessage(logString, data.lang);
    maybeSetFruit(data.message);
    connection.send('Received message ' + messageIdx);
});
```

Source: https://googlechrome.github.io/samples/presentation-api/index.html
HTML 5.2

Features removed (some …)

- keygen, menu and menuitem (added in 5.1) elements
- inputmode attribute for textual input elements, and the dropzone attributes
- showModalDialog method
HTML 5.2

Constructions now valid HTML

- `<style>` within the `<body>`
- multiple `<main>` elements in the DOM, so long as only one is visible to the user
  - The main content area of a document includes content that is unique to that document and excludes content that is repeated across a set of documents such as site navigation links, copyright information, site logos and banners and search forms (unless the document or application’s main function is that of a search form).

why? Accessibility ...
Index

- Introduction
- API
- Evolution
- Hands on
- Conclusions
Hands on Image Editor

Canvas, Drag Files, Local Storage
Hands on Service Workers

Performance

Félix Albertos Marco

ICWE 2018 TUTORIAL

Review of the HTML5 API
Hands on Service Workers

Taking the Control
What

• Proxy servers that sit between web applications, the browser, and the network
How

• Intercept network requests and take appropriate action based on whether the network is available, updating assets residing on the server*
How

• Intercept network requests and take appropriate action based on whether the network is available **or not**, assets locally cached can be used.
Enable the creation of effective offline experiences.
Scenarios

- Updating assets from the server
Scenarios

• Working offline

Hands on Service Workers
Scenarios

• Check if asset is cached (yes)
Scenarios

• Check if asset is cached (no)
Scenarios

- Check if asset is cached (no)
Lifecycle

- Lifecycle
Lifecycle

• Register Service Workers in the Web Application

```html
<script>
  navigator.serviceWorker.register('serviceWorker.js')
  .then(reg => console.log('SW registered!', reg))
  .catch(err => console.log('SW not registered', err));
</script>
```
Attention

- You don’t have access to window objects
- Service worker file run only over HTTPS server or your localhost
- Service worker file must be registered at the root directory of your website
Lifecycle

✗ (Configuring the Service Worker)

Installing
The SW is registered in the browser for that site

Waiting
The SW is waiting for closing all the clients

✓ Active
Precious SW (if exists) is gone, new SW is able to control clients
### Lifecycle

- Some configuration for the Web Application

```javascript
var BASEdir = '/icwe18tut/ServiceWorkers/
var cacheName = 'v0'.
var offlineHTML = BASEdir+'content-not-available.html';
var offlineIMG = BASEdir+'img/OfflineImage.png';

// Fixed, Incremental, Free --> Schemas for offline navigation
var incrementalSchema = true;

// Assets
var BASEdir = '/icwe18tut/ServiceWorkers/';
var cacheName = 'v0';
var offlineHTML = BASEdir+'content-not-available.html';
var offlineIMG = BASEdir+'img/OfflineImage.png';

function registerServiceWorker() {
  navigator.serviceWorker.register('/icwe18tut/ServiceWorkers/v0').then(function(reg) {
    console.log('Service Worker registered!', reg)
  }).catch(function(err) {
    console.log('Service Worker failed to register', err)
  })
}

registerServiceWorker();
```

<table>
<thead>
<tr>
<th>Schema</th>
<th>Available elements</th>
<th>Initial elements</th>
<th>Does it take into account user actions?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed</td>
<td>Fixed</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Free</td>
<td>Variable</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Incremental</td>
<td>Fixed + Variable</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Lifecycle

• Events
  • install

```javascript
self.addEventListener('install', function(e){
  console.log("[ServiceWorker] Installed");

  // IMPORTANT !!!
  // Avoid waiting for closing all the clients
  // Remove if data consistence is important !!!
  // self.skipWaiting(); // <--------------------------------------

  /////////////////////////////////////////////////////////////

  e.waitUntil(
    caches.open(cacheName).then(function(cache){
      console.log("[ServiceWorker] Caching cacheFiles");
      return cache.addAll(cacheFiles);
    }).catch(error =>{
      console.log('ERROR: ' +error);
    })
  
 ));
```

Register
Configuration
Installing
Waiting
Active
Fetch
Lifecycle

- Events
  - activate

```javascript
self.addEventListener('activate', function(e){
  console.log("[ServiceWorker] Activated");

  e.waitUntil(
    caches.keys().then(function(cacheNames){
      return Promise.all(cacheNames.map(function(thisCacheName){
        if(thisCacheName !== cacheName){
          console.log("[ServiceWorker] Removing Cached Files from", thisCacheName);
          return caches.delete(thisCacheName);
        }
      })
    )
  ));
});
```
Lifecycle

- Events
  - fetch

```javascript
self.addEventListener('fetch', function(event) {
  console.log('[ServiceWorker] Fetch', event.request.url);
  event.respondWith(
    caches.match(event.request)
      .then(function(response) {
        if(response) {
          console.log("From cache");
          return response;
        } else {
          console.log("From network");
          return fetch(event.request).then(function(response) {
            if(incrementalSchema) {
              caches.open(cachename).then(function(cache) {
                cache.put(event.request, response.clone());
              });
            }
            return response.clone();
          }).catch(error ->{
            var ext = event.request.url.split('.').pop();
            if(["html","htm","php"].indexOf(ext) >= 0) {
              return caches.match(offlineHTML);
            } else if(["png","jpg"].indexOf(ext) >= 0) {
              return caches.match(offlineIMAGES);
            } else {
              return caches.match(offlineHTML);
            }
          });
        }
      });
});
```
Lifecycle

• Dive into the fetch Event
Scenarios

- Check if asset is cached (yes)
Scenarios

• Check if asset is cached (no)

```javascript
self.addEventListener('fetch', function(event) {
  console.log('[ServiceWorker] Fetch', event.request.url);
  event.respondWith(
    caches.match(event.request)
      .then(function(response){
        if(response){
          console.log("from cache");
          return response;
        }else{
          console.log("from network");
          return fetch(event.request).then(function(response) {
            if(incrementalSchema){
              caches.open(cacheName).then(function(cache) {
                cache.put(event.request, response.clone());
              });
            }
            return response.clone();
          }).catch(error =>{
            var ext = event.request.url.split('.').pop();
            if(['html','htm','php'].indexOf(ext)==0){
              return caches.match('offlineHTML');
            }else if(['png','jpg'].indexOf(ext)==0){
              return caches.match('offlineIMG');
            }else{
              return caches.match('offlineHTML');
            }
          });
        }
      });
});
```
Scenarios

• Check if asset is cached (no)
Lifecycle

- Debugging tool (Chrome)
  - Chrome
- Application
  - Application
    - Service Workers
- Application
  - Cache
    - Cache Storage
Service Workers

• Lessons learned
  • Be aware of the Browsers cache
  • `<ctrl><f5>` updates cache ... or not
    • `<f5>` shows previous version
  • `shift+reload` a document (Debugger tools), it’ll always load without a controller, which is handy for testing quick CSS & JS changes
  • Only updates when browser closes completely
    • But, `self.skipWaiting()`!
    • `Service Workers work as native applications` --> close the browser completely for update them
  • Code Consistency and Data consistency
    • Different browser tabs may run different code --> `corrupt client-side database`
Service Workers

• Lessons learned
  • event.request
    • Be careful when using it
      • console.log(event.request) ?
    • Only one use
      • Clone
  • Caches.match vs fetch
    • When offline always caches.match (avoid the problems I have for that mistake ;) )
• Reload the Web application to use the new version
Service Workers

• The secret life of a Service Worker
  • Service Worker do not ends when the browser is closed
    • Variables ...
  • But if the computer is restarted, the service worker starts again
Service Workers

• *Bonus
  • Communication between the Web Application and the Service Worker
    • Using MessageChannel
  • Push notifications
  • Background sync API
MessageChannel

- Part of the Channel Messaging API
  - Allows the creation of a new message channel
  - Send / Receive data through it via its two MessagePort properties

```javascript
class sMessageChannel{
  constructor()
  this._channel = null;
  this._port1 = null;
  this._port2 = null;
  this._command = null;
  this._data = null;
}

_init(){
  this._channel = new MessageChannel();
  this._port1 = this._channel.port1;
  this._port2 = this._channel.port2;

  this._data = {
    command: this._command,
    msg: this._message
  };
}

sendMessage(v){
  navigator.serviceWorker.controller.postMessage(v, [this._port2]);
}

exec(v){
  this._message = v;
  this._init();
  var _this = this;
  return new Promise(function(resolve, reject){
    _this._port1.message = function(e){
      if(e.data.error){
        reject(e.data.error);
      }else{
        resolve(e.data);
      }
    };

    _this.sendMessage(JSON.stringify(_this._data));
  });
})
```

Client Side
MessageChannel

Client Side

Service Workers Side
MessageChannel

• Execution
  • Creation of the MessageChannel
    • (using the previous class)
  • Send the message
    • (exec method)
  • Read the result
    • PromiseValue

```javascript
var test = new chkFileOnCache();
if (undefined)
  test.exec("c1.html");
  Promise {<pending>}
    __proto__: Promise
      [[PromiseStatus]]: "resolved"
      [[PromiseValue]]: true
  test.exec("c2.html");
  Promise {<pending>}
    __proto__: Promise
      [[PromiseStatus]]: "resolved"
      [[PromiseValue]]: false
```

Execution on the Console
Execution on a Web Application
Push Notifications

• Push Notifications
  • The Notifications API displays notifications to the user
  • It is incredibly powerful and simple to use
  • Where possible, it uses the same mechanisms a native app would use, giving a completely native look and feel
Push Notifications

- Push Notifications
  - First request permission for sending notifications
    - Notification.requestPermission
  - If granted
    - showNotification method to send notifications

```javascript
Notification.requestPermission(function(result) {
  if (result === 'granted') {
    navigator.serviceWorker.ready.then(function(registration) {
      registration.showNotification('ICWE 2018 Registration', {
        body: 'Please, complete the registration process.',
        vibrate: [200, 100, 200, 100, 200, 100, 200],
        icon: 'img/logobig.png',
        tag: 'Registration to ICWE 2018',
        data: {
          dateOfArrival: Date.now(),
          primaryKey: 'registration'
        }
      });
    });
  }
});
```
Push Notifications

• Case Study 1
  • Show an invitation to the user for participating in a conference
Push Notifications

• Case Study 1
  • Show an invitation to the user for participating in a conference
Push Notifications

- Case Study 2
  - Remember the user to **complete** the registration process
Push Notifications

• Case Study 2
  • Remember the user to complete the registration process

Registration Form

First name: 
Last name: 
Submit

You are not registered !!!

Registration Form

First name: John
Last name: Doere
Submit

Thank you for your registration :)

ICWE 2018 Registration

Please, complete the registration process.

localhost
Index

- Introduction
- API
- Evolution
- Hands on
- Conclusions
Conclusions

Attention

You don't have access to window objects
Service worker file ran only over HTTPS server or your localhost
Service worker file must registered at the root directory of your website

• Check if asset is cached (no)
Conclusions

• Why, When, How
Mastropiero Terrace

one of the most interesting terraces in Spain to have some "tapas"
Review of the HTML5 API

Computer Systems Department, University of Castilla-La Mancha
Albacete, Spain
felix.albertos@uclm.es

5th June 2018