



2016

BADWORK (control de acceso)



Carlos Cañas Agenjo

IES Virgen de la Paz

08/06/2016

Índice

1. Justificación del proyecto y objetivos.....	3
2. Introducción	4
3. Metodología y desarrollo del proyecto.....	5
3.1. Metodología	5
3.2. Desarrollo del proyecto	6
3.2.1. Ciclo 0.....	6
3.2.1.1. Comunicación con el cliente / Objetivo.	6
3.2.1.2. Planificación.....	6
3.2.1.2.1. Diagrama gráfico de Gantt	7
3.2.2. Ciclo 1. Liberación de BadWork 0.0.1.....	8
3.2.2.1. Comunicación con el cliente / Objetivo.	8
3.2.2.2. Planificación.....	8
3.2.2.2.1. Diagrama gráfico de Gantt	8
3.2.2.3. Análisis de riesgo	9
3.2.2.3. Ingeniería.....	9
3.2.2.4. Evaluación / Casos de uso y pruebas.....	15
3.2.2.5. Construcción y entrega	16
3.2.3. Ciclo 2. Liberación de BadWork 1.0.1.....	17
3.2.3.1. Comunicación con el cliente / Objetivo.	18
3.2.3.2. Planificación.....	18
3.2.3.2.1. Diagrama gráfico de Gantt	18
3.2.3.3. Análisis de riesgo	19
3.2.3.4. Ingeniería.....	19
3.2.3.4. Evaluación / Casos de uso y pruebas.....	26
3.2.3.5. Construcción y entrega	29
4. Resultado y discusión.....	34
5. Conclusiones	35
6. Bibliografía y referencias	36
7. Anexos/Otros.....	37
7.1. Anexo I.....	37
7.2. Anexo II.....	38
7.3. Anexo III.....	43

1. Justificación del proyecto y objetivos.

Los navegadores populares Chrome y Firefox nos dan la posibilidad de añadir mejoras y personalizarlos al uso cotidiano de cada usuario, esta característica es lo que me ha llevado a la elección de este proyecto.

Estos programas que instalamos en nuestros navegadores, hacen de ellos una herramienta muy potente debido a que no sólo realizan su uso funcional, sino que la optimizan y pueden tener diferentes usos y características.

El desarrollo de esta aplicación lo realizaré para el navegador Mozilla Firefox ya que comparto su filosofía de que el conocimiento debe ser libre, gratuito y debe ser difundido para toda la comunidad del mundo.

El proyecto que voy a realizar será un control de acceso, centrado en la actividad laboral de una empresa, cuya función es que el navegador bloquee las páginas no deseadas por el administrador.

Mi elección ha sido debida a que en los últimos años se ha visto un aumento de bajo rendimiento laboral, debido a la libertad que nos da internet, lo que conlleva a que los empleados hagan un mal uso de las nuevas tecnologías y en sus horas de actividad profesional no sólo accedan a contenido relacionado con el desarrollo de su profesión, sino también a contenido personal y de ocio.

Los objetivos que me planteo para dicho proyecto son los siguientes:

- Principales:
 1. Bloquear las páginas no deseadas por el administrador.
 2. Gestionar las páginas bloqueadas.
- Secundarios:
 1. Prevenir el mal uso de las nuevas tecnologías.
 2. Aumentar el rendimiento de los empleados, por no tener acceso libre a internet.

2. Introducción

El control de acceso es una herramienta que se puede habilitar por parte del administrador de forma rápida y sencilla, además personalizarlo según las necesidades de cada usuario.

Su funcionalidad es filtrar el contenido web y la actividad de registro.

La habilitación y deshabilitación de dicho control de acceso normalmente está bloqueada por una contraseña, que únicamente conoce el administrador.

Las principales características del control de acceso son:

- Bloqueo de páginas web: restringe el acceso a contenido inapropiado por medio del bloqueo de ciertos sitios web.
- Restricción de acceso por categorías: aparecen en conjuntos, se muestra un listado donde se selecciona la categoría deseada a bloquear.
- Registro de actividad: permite observar un registro detallado de la actividad dentro del control de acceso.
- Selección rápida: se crean usuarios y según cada rol, se le permite el acceso a determinado contenido o no. Ejemplo: niño, adolescente, padre.
- Control de tiempo: limita el tiempo del ordenador conectado a internet.

El control de acceso es aplicable a diversos campos:

- Control en el hogar familiar, dónde los padres serían los administradores.
- Control escolar, en los centros educativos, bloqueando ciertos contenidos inapropiados e innecesarios durante las horas lectivas.
- Control laboral, en las empresas, para que así los empleados no hagan un mal uso de las nuevas tecnologías ni abusen del libre acceso por internet.

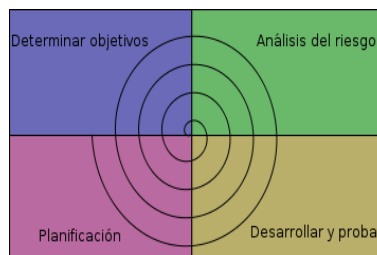
Mi proyecto de control de acceso se va a centrar en un control laboral, dentro de los ordenadores de una empresa, realizando un bloqueo de páginas web, para que los empleados puedan acceder sólo a los contenidos necesarios que se relacionen con su desarrollo laboral y así conseguir un aumento de su rendimiento. Se llamará “BADWORK” (mala palabra).

3. Metodología y desarrollo del proyecto

3.1. Metodología

La metodología que voy a seguir para desarrollar este proyecto será un modelo en espiral.

Cada bucle representa un conjunto de actividades, esto formará una caracola, que tiene dos dimensiones, la angular (indica el avance del proyecto) y la radial (indica el aumento de costes, porque por cada nueva interacción quiere decir que se pasa más tiempo desarrollando).

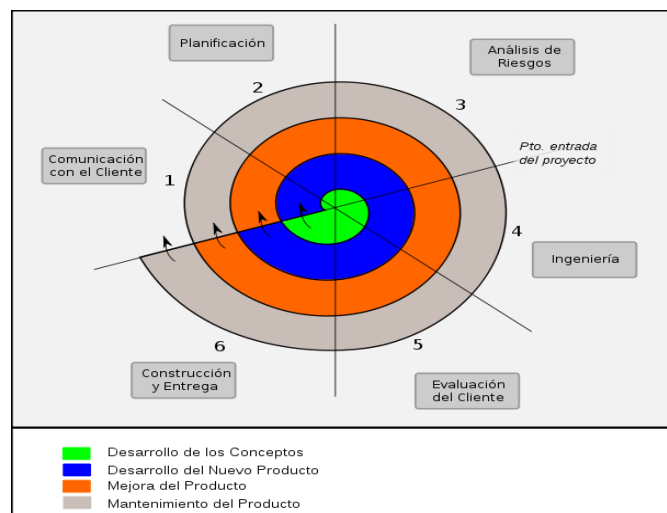


Cada ciclo tendrá:

- Objetivos.
- Análisis de riesgos y alternativas.
- Desarrollo y prueba.
- Planificación, donde se probará finalmente y se verá si en necesario comenzar un nuevo ciclo.

El modelo en espiral se divide en regiones de tareas:

- Comunicación con el cliente.
- Planificación.
- Análisis de riesgos.
- Ingeniería: tareas para construir representaciones de la aplicación.
- Evaluación: casos de usos y pruebas.
- Construcción y entrega.



3.2. Desarrollo del proyecto

3.2.1. Ciclo 0.

En este ciclo se abordará el estudio de la tecnología XUL. Este ciclo se tomará como una fase de estudio previo al desarrollo del proyecto.

3.2.1.1. Comunicación con el cliente / Objetivo.

- El cliente solicita que se realice un control de acceso para su empresa, su intención es que los empleados no puedan acceder a determinadas páginas web.

Objetivo: bloquear y desbloquear la página actual, y que esto quede guardado en la sesión.

3.2.1.2. Planificación.

XUL (XML Lenguaje de Interfaz de Usuario)

Esta tecnología fue creada para facilitar el desarrollo del navegador Mozilla Firefox. Es un lenguaje XML (eXtensible Markup Language), por lo que todas las características de XML también están incluidas en XUL.

La mayoría de aplicaciones necesitan ser desarrolladas usando una plataforma específica, convirtiendo el desarrollo de la misma en algo costoso. Sin embargo XUL permite ser implementada y modificada de forma fácil y rápida.

XUL tiene las ventajas de otros lenguajes XML como XHTML (eXtensible HyperText Markup Language) y también puede aplicar hojas de estilos para modificar la apariencia de la interfaz del usuario.

Estos son algunos elementos que pueden ser creados:

- Controles de entrada tales como cuadros de texto y cajas de chequeo.
- Barra de herramientas con botones u otros contenidos.
- Menús en barras de menú o menú emergente.
- Pestañas de diálogo.
- Árbol de información jerárquica o tabulada.
- Teclas de accesos directo.

Para desarrollar en XUL el usuario debe tener conocimientos de HTML (HyperText Markup Language), XML y CSS (Cascading Style Sheets).

Reglas que se deben tener en cuenta:

- Los elementos en XUL y sus atributos deben introducirse en minúsculas, ya que XML distingue entre mayúsculas/minúsculas (a diferencia de HTML).

- Los valores de los atributos en XUL, deben colocarse entre comillas, aunque sean números.
- Los archivos XUL por lo general se dividen en cuatro ficheros, uno para la disposición de los elementos, otro para la definición del estilo, otro para declarar las entidades (usadas en las localizaciones) y otro para los "script". Además se pueden tener archivos para las imágenes o para datos específicos de una plataforma.

Para crear fácilmente el desarrollo de aplicaciones con esta tecnología es necesario instalar:

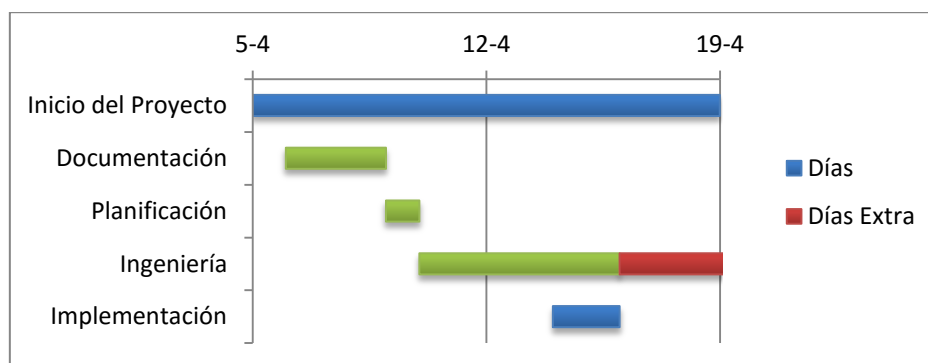
- IDE (integrated development environment) “entorno de desarrollo interactivo”: XUL Explorer, facilita la creación de ficheros con extensión “.xul”.
- Remote xul manager (extensión de Firefox para desarrollar y probar ficheros XUL).

Justificación de cambio de tecnología

Comienzo a desarrollar el proyecto con la tecnología XUL, y me encuentro con varias dificultades que lo hacen incompatible, por lo que cambio a una tecnología más actual llamada ADD-ONS SDK (Software Development Kit). Este cambio es comentado y aceptado por el cliente ya que conlleva un retraso en la planificación del proyecto.

3.2.1.2.1. Diagrama gráfico de Gantt

Tareas	Fecha de inicio Programada	Fecha de inicio	Días	Días Extra	Fecha fin
Inicio del Proyecto	05/04/2016	05/04/2016	14	0	19/04/2016
Documentación	06/04/2016	06/04/2016	13	0	19/04/2016
Planificación	09/04/2016	09/04/2016	1	0	09/04/2016
Ingeniería	10/04/2016	10/04/2016	4	6	19/04/2016
Implementación	11/04/2016	14/04/2016		5	19/04/2016



3.2.2. Ciclo 1. Liberación de BadWork 0.0.1

En este ciclo se hará una aplicación para bloquear y desbloquear páginas web, y se realizará la entrega con una aplicación firmada y preparada para instalar en nuestro navegador Firefox.

3.2.2.1. Comunicación con el cliente / Objetivo.

- El cliente solicita que realice un control de acceso para su empresa, su intención es que los empleados no puedan acceder a determinadas páginas web.

Objetivo: bloquear y desbloquear la página actual, y que esto quede guardado en la sesión.

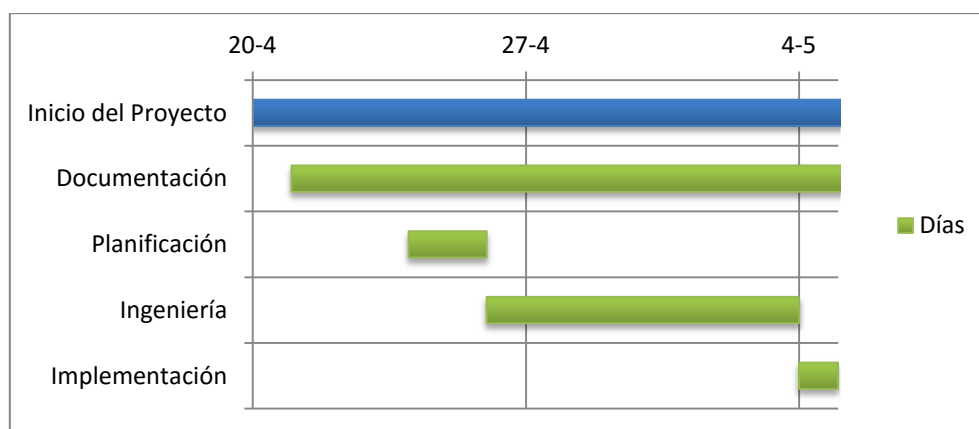
3.2.2.2. Planificación.

ADD-ONS SDK

Add-on SDK permite crear complementos Firefox utilizando las tecnologías Web estándar: JavaScript, HTML y CSS. SDK incluye una API de JavaScript que se puede utilizar para crear complementos y herramientas de creación, funcionamiento, pruebas y empaquetado, llamada JPM que será descrita en el apartado de ingeniería.

3.2.2.2.1. Diagrama gráfico de Gantt

Tareas	Fecha de inicio Programada	Fecha de inicio	Días	Días Extra	Fecha fin
Inicio del Proyecto	20/04/2016	20/04/2016	16	0	05/05/2016
Documentación	21/04/2016	21/04/2016	15	0	05/05/2016
Planificación	24/04/2016	24/04/2016	2	0	25/04/2016
Ingeniería	26/04/2016	26/04/2016	8	0	03/05/2016
Implementación	04/05/2016	04/05/2016	1	0	05/05/2016



3.2.2.3. Análisis de riesgo

Llevo a cabo el estudio de las posibles amenazas y eventos no deseados, evaluando y proponiendo diferentes alternativas:

- Al buscar información sobre los controles de acceso y el tipo de tecnología usada habitualmente, una de las dificultades que se me presentan es que la mayoría de la documentación viene en lengua extranjera. Por lo que veo el idioma como un posible riesgo. Aspecto solucionable al poder utilizar software para traducir idiomas.
- Otro de los riesgos que me planteo es encontrarme con versiones no actualizadas de la tecnología a utilizar para desarrollar el proyecto, como alternativa a esto me propongo intentar realizar el proyecto con la tecnología más óptima.
- Retraso en los plazos de entrega, por la necesidad de solucionar diferentes riesgos. Decido añadir algún día de margen al plazo de entrega.

3.2.2.3. Ingeniería

Para desarrollar el proyecto usaré las siguientes herramientas:

- Editor de texto Sublime Text.
- Navegador Firefox con las siguientes extensiones instaladas: firebug, colorzilla.

Para crear complementos para Firefox utilizando el SDK, tenemos una serie de requisitos:

- Navegador Firefox en la versión 38 o superior.
- Tener instalado Node.js.
- Tener instalado el paquete JPM.

JPM es distribuido por un gestor de paquetes nodo (NPM) que podemos conseguir a través de Node.js.

Una vez instalado Node.js ejecutaremos los siguientes comandos en nuestra terminal:

- **git clone <https://github.com/mozilla-jetpack/jpm.git>**
- **cd jpm**
- **npm install**
- **npm link**

La herramienta JPM es el reemplazo para CFX . JPM permite probar, ejecutar, y crear paquetes de complementos a través de la línea de comandos. Estos son algunos de los comandos que puede utilizar:

jpm init	Crear una estructura del add-ons a desarrollar.
jpm run	Lanza una instancia a Firefox con el complemento instalado.
jpm test	Ejecuta las pruebas de la unidad del complemento.
jpm xpi	Empaqueta todos los archivos del complemento en XPI, que es el formato de archivos para instalar complementos Firefox.
jpm post	Empaqueta los archivos en XPI y les asigna una ruta.
jpm watchpost	Empaqueta los archivos en XPI cada vez que hay cambios en el archivo y lo publica en cierta URL.
jpm sign	Empaqueta los archivos en XPI y valida para posteriormente firmar el complemento por Mozilla.

En Add-ons SDK hay dos grupos de APIs/módulos que sirven para desarrollar:

- APIs de alto nivel: son aquellas que sirven para la construcción de complementos como: creación de interfaz del usuario, interactuando con la web, e interactuando con el navegador. Éstas son las APIs de alto nivel que se utilizarán para liberar esta versión:
 - simple-storage
 - self
 - tabs
 - panel
- APIs de bajo nivel: estos módulos todavía están en desarrollo, por lo que se espera que no sean utilizados en futuras versiones. Se dividen en módulos para poder crear colecciones, módulos para crear eventos y trabajadores, y por último módulos que sirven como complemento de APIs de alto nivel. De este grupo se van a usar los siguientes módulos:
 - ui/button/toggle

Para utilizar las diferentes APIs primero hay que declarar una variable igualándola a la librería requerida:

```
const data = require("sdk/self").data;
const toggleButtons = require('sdk/ui/button/toggle');
const tabs = require("sdk/tabs");
const panels = require("sdk/panel");
const urls = require("sdk/url");
const ss = require("sdk/simple-storage");
```

Explicación de los módulos utilizados:

Simple-storage: permite generar un almacén de datos con persistencia, quedando guardado en la memoria del perfil del navegador Firefox. Funciona de forma similar al almacenamiento del DOM en la Web (local.storage), con la excepción de que sólo está disponible para estos complementos.

Datos que puede almacenar:

- strings
- boolean
- números
- arrays
- objetos
- nulos

Self: Este módulo permite acceder a los datos que hay dentro del propio add-on. No se debe confundir con el objeto global **self** que se utiliza en los scripts de contenido para comunicarse con el código añadido.

Métodos:

Self.data.url (nombre): apunta a un fichero de datos incrustado, muy útil para visualizar el contenido de un fichero HTML que puede ir incrustado, por ejemplo, en el constructor de un panel.

Tabs: abrir, manipular, recibir mensajes y eventos de las pestañas. En esta versión se usará esta API por sus diferentes propiedades:

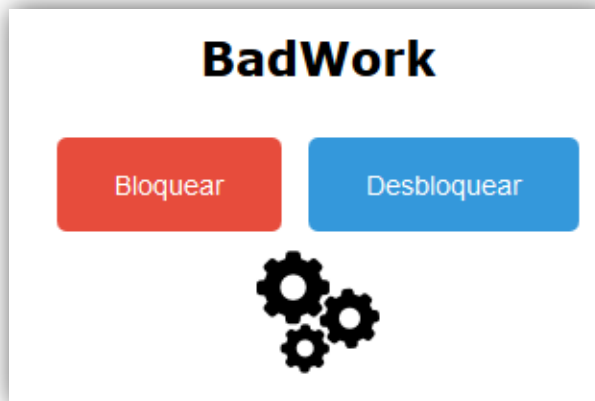
- **activeTab:** llama al objeto pestaña activado.
- **url:** devuelve o carga una url de la pestaña activa.

Panels: crea diálogos transitorios que forman parte de la interfaz del complemento.

Constructor:

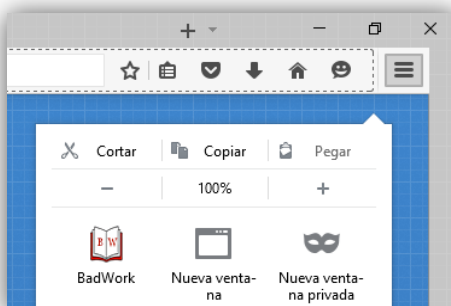
Opciones	Descripción
With	Ancho del panel. Valor numérico.
Heigth	Alto del panel. Valor numérico.
contentURL	Con el módulo self podremos inyectar en el contenido del panel un fichero HTML ya guardado en la extensión.
onHide	Incluye una función para escuchar eventos ocultos del panel.

```
// constructor del panel principal
var panel = panels.Panel({
  width: 350,
  height: 220,
  contentURL: self.data.url("panel.html"),
  contentScriptFile: self.data.url("panel.js"),
  onHide: handleHide
});
```



Ui/button/toggle: Sirve para agregar un botón de activación a la interfaz del usuario de Firefox. Con este módulo puedes crear botones que funcionan como una casilla de verificación, como si representara un interruptor.

En el constructor de este módulo podremos determinar los diferentes tamaños de los iconos, que representarán nuestra extensión, y asignarle un evento que al pulsarlo se active. Por ejemplo que se muestre el panel construido anteriormente.

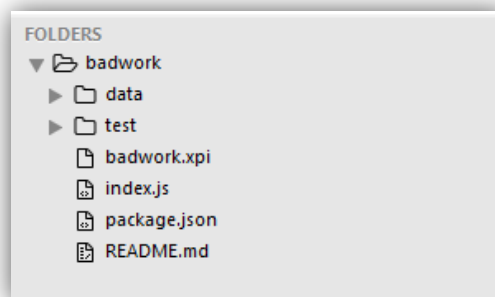


```
// constructor del boton principal
var button = toggleButtons.ToggleButton({
  id: "my-button",
  label: "BadWork",
  icon: {
    "16": "./icon-16.png",
    "32": "./icon-32.png",
    "64": "./icon-64.png"
  },
  onChange: handleChange
});
```

Para empezar el desarrollo lo primero que hago es pensar un nombre para el programa, ya que el objetivo es evitar distracciones en el empleo creo que una buena posibilidad es llamarlo “BADWORK”, el símbolo elegido para representarlo es el siguiente:



Con el comando **JPM INIT** he creado una estructura base, a la que podemos añadir diferentes cambios y explicar los diferentes directorios y archivos:



Badwork: carpeta raíz.

Data: carpeta dónde se van almacenar las imágenes, JS, HTML y CSS.

Test: carpeta que va a almacenar ficheros JS para hacer pruebas.

Badwork.xpi: será el fichero compilado preparado para instalar.

Index.js: será el fichero que llevará la lógica de la API.

Package.json: archivo que describe características de la API, autor, colaborador, versión soportada, descripción, nombre, etc.

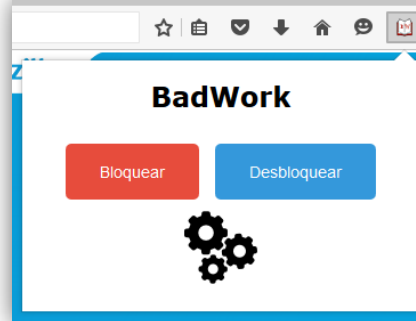
README.md: pequeña descripción del funcionamiento de la API.

Almaceno en la carpeta data tres imágenes con distinto tamaño:

- Icon-16.png
- Icon-32.png
- Icon-64.png



En adelante se crean los diferentes archivos HTML y CSS para dar estilo al panel. Se codifica la lógica en el index.js y se construyen los diferentes objetos, de los que he hablado con anterioridad, el panel y el botón de activación, quedando con el resultado siguiente:

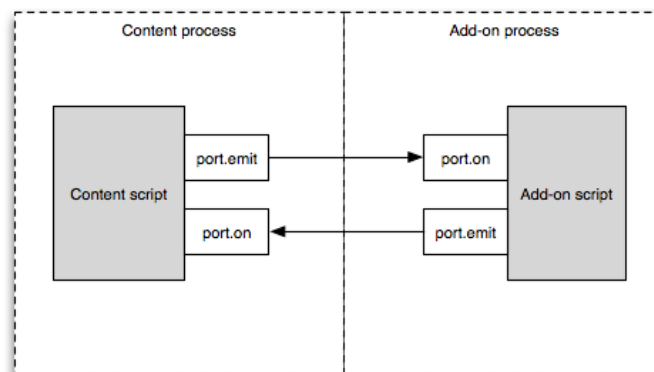


Para habilitar la comunicación entre sí de las secuencias de comandos de add-ons y las secuencias del contenido de un módulo add-on SDK, se tiene que crear un puerto objeto:

- Para enviar mensajes se usa `port.emit()`
- Para recibir el mensaje del otro lado se usa `port.on()`

Estos mensajes son asíncronos, es decir, no se espera una respuesta del receptor, una vez enviado el mensaje el script sigue su procesamiento.

Para comunicarse entre sí las secuencias de comandos y secuencias de comandos de contenido del add-ons, cada extremo de la conversación tiene que habilitar el acceso a su puerto objeto. A continuación se muestra un esquema y un ejemplo:

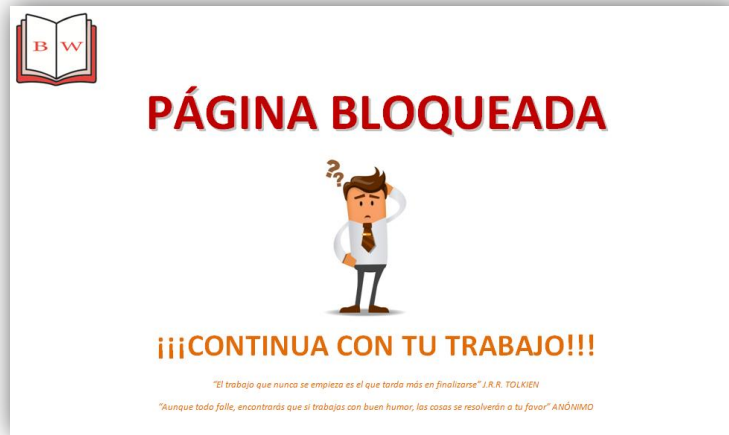


```
// boton que envia al index para bloquear la url
btnBlocked.addEventListener('click', function onkeyup(event){
  self.port.emit('blocked');
},false);

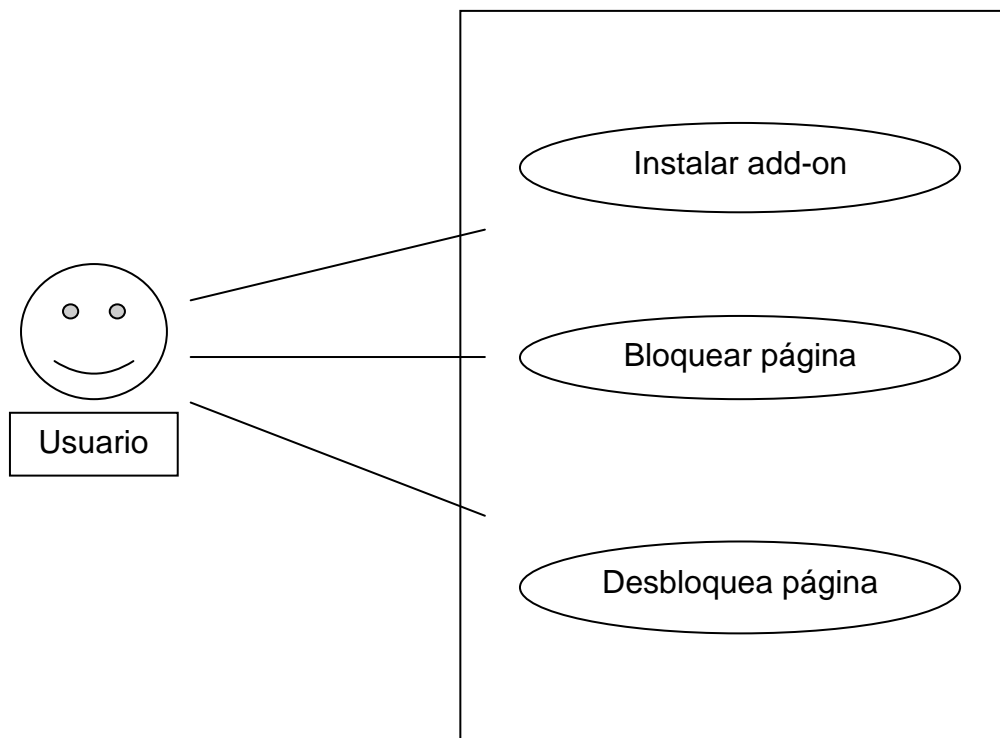
// boton que envia al index para desbloquear la url
btnUnblocked.addEventListener('click', function onkeyup(event){
  self.port.emit('unblocked');
},false);
```

En este ejemplo vemos como se generan dos mensajes emit y se enviará uno de ellos dependiendo del botón pulsado del panel.

Cuando el index.js recoja el mensaje de bloqueo, almacenará la ruta y redireccionará la página al fichero 'blocked.html' quedando de la siguiente manera.



3.2.2.4. Evaluación / Casos de uso y pruebas



Nombre	UC-01 bloquear página.
Secuencia principal	<ol style="list-style-type: none"> 1. El usuario solicita bloquear www.mozilla.org. 2. El sistema recoge la información de la url. 3. El sistema almacena el dominio en un array. 4. El sistema redirecciona a la página de bloqueo.
Error / Secuencia alternativa	<p>1.1.i. El usuario solicita abrir www.mozilla.org (ya bloqueada) en una nueva pestaña.</p> <p>1.1.p La página está bloqueada.</p> <p>1.2.i. El usuario abre varias pestañas con la url www.mozilla.org y solicita en una pestaña el bloqueo de la página.</p> <p>1.2.p. ERROR. Sólo se bloquea la página activa, las demás pestañas con la misma url no se han bloqueado.</p> <p>1.2.i. El usuario solicita desbloquear www.mozilla.org. Caso de uso finaliza y comienza UC-02 desde la secuencia 2.</p>
Post Codificación	El usuario ve que la página está bloqueada.

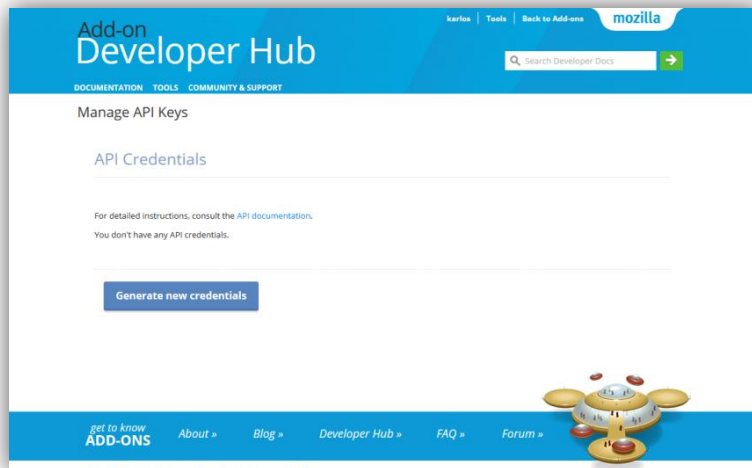
Nombre	UC-02 desbloquear página.
Secuencia principal	<ol style="list-style-type: none"> 1. El usuario solicita desbloquear www.mozilla.org. 2. El sistema encuentra la página para desbloquear. 3. El sistema borra el dominio del array. 4. El sistema redirecciona a la página www.mozilla.org.
Error / Secuencia alternativa	<p>1.1.i. El usuario abre varias pestañas con la url www.mozilla.org, pagina bloqueada, y solicita desbloquear en una de ellas.</p> <p>1.1.p. ERROR. Sólo se desbloquea la pestaña activa. Las demás pestañas siguen bloqueadas.</p>
Post Codificación	El usuario vuelve a ver la página.

3.2.2.5. Construcción y entrega

Para conseguir construir nuestra add-on es muy sencillo, sólo basta con usar el comando **JPM XPI** y se comprimirá toda la aplicación. Este archivo podremos instalarlo en el navegador arrastrándolo sobre él. Pero esta técnica conlleva un problema, el navegador sabe que esa aplicación no está firmada por los agentes de Mozilla y te advierte que no es segura.

En esta versión voy a mostrar cómo obtener la firma de los agentes de Mozilla de forma sencilla.

- Esta dirección nos proporcionará un usuario y una clave privada:
<https://addons.mozilla.org/en-US/developers/addon/api/key/>



- En nuestra línea de comandos ejecutaremos lo siguiente:
jpm sign --api-key \${AMO_API_KEY} --api-secret \${AMO_API_SECRET}
- Pasa una serie de validaciones que pueden dar los siguientes mensajes:
 - El proceso de validación fue correcto, Mozilla firmó el proyecto y generó un archivo XPI firmado.
 - El proceso de validación fue fallido y no se firmó el complemento.
 - El número de versión de su complemento ya existe. Incrementa el número de la versión en el package.json y vuelve a ejecutar el comando.

Si todo ha funcionado ya poseemos nuestro complemento firmado y lo podremos instalar en nuestro navegador. La peculiaridad que tiene JPM para que firme Mozilla, es que el completo está en los repositorios de Mozilla, pero en una zona no listada, por lo que para distribuir la aplicación deberemos hacerlo nosotros mismos.

3.2.3. Ciclo 2. Liberación de BadWork 1.0.1.

Este ciclo será el último para la entrega del proyecto. Se realizarán muchos cambios, tanto a nivel de diseño como de codificación, por lo que pasará por varias versiones hasta llegar a la final. Este ciclo termina pudiendo descargar la aplicación desde los complementos de Firefox e instalarlo en nuestro navegador.

3.2.3.1. Comunicación con el cliente / Objetivo.

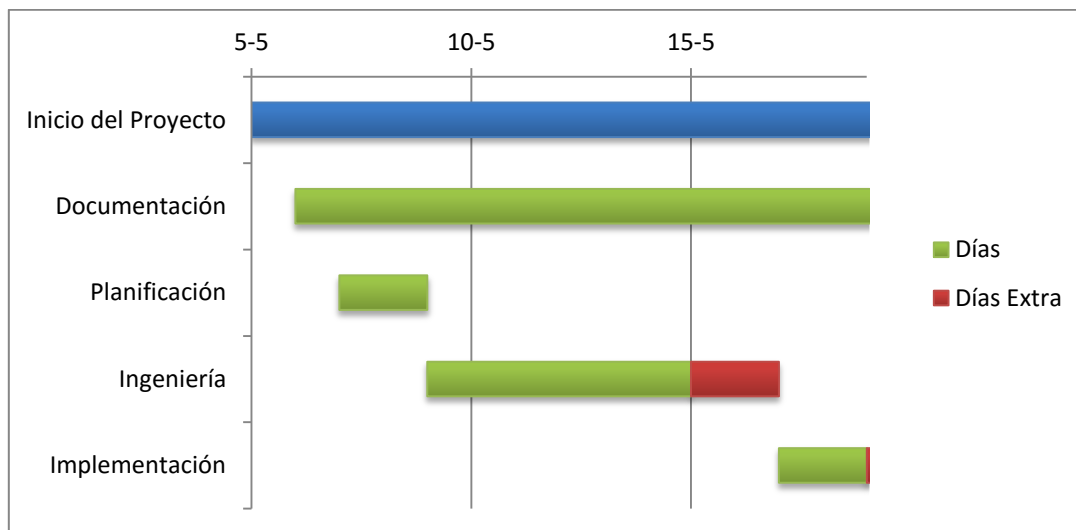
- El cliente quiere añadir varias funcionalidades más:
 1. Una contraseña de acceso para que sólo el administrador pueda añadir o quitar páginas bloqueadas.
 2. Una plataforma de configuración dónde se listan las páginas bloqueadas y la posibilidad de cambiar la contraseña desde esta zona.

Objetivo: crear una mayor seguridad en la aplicación para que sólo el administrador pueda añadir o borrar las páginas que desea bloquear.

3.2.3.2. Planificación.

3.2.3.2.1. Diagrama gráfico de Gantt

Tareas	Fecha de inicio Programada	Fecha de inicio	Días	Días Extra	Fecha fin
Inicio del Proyecto	05/05/2016	05/05/2016	15	0	19/05/2016
Documentación	06/05/2016	06/05/2016	14	0	07/05/2016
Planificación	07/05/2016	07/05/2016	2	0	08/05/2016
Ingeniería	09/05/2016	09/05/2016	6	2	16/05/2016
Implementación	16/05/2016	17/05/2016	2	1	19/05/2016



3.2.3.3. Análisis de riesgo

Llevo a cabo el estudio de las posibles amenazas y eventos no deseados, evaluando y proponiendo diferentes alternativas:

- Utilizar contraseñas sin cifrado, cabe la posibilidad de que una tercera persona quiera hacer un mal uso de la aplicación. Como solución todas las contraseñas serán cifradas con el algoritmo SHA256.
- Incomunicación entre los diferentes ficheros javascript al pasar variables entre ellos. Para solventarlo utilizaré funciones de llamadas de mensajes provistas por el propio lenguaje.
- Pérdida de datos del proyecto por daños en el sistema o en los ficheros. Como solución, crearé un repositorio GitHub para mantener un control de versiones y asegurar el funcionamiento correcto de cada versión.

3.2.3.4. Ingeniería

En esta versión utilizaré el control de versiones GitHub.

En esta tabla se muestran algunos comandos que he utilizado:

Git clone	Clonar el repositorio.
Git status	Saber el estado de los ficheros locales frente a los del repositorio.
Git diff	Ver la diferencia de los cambios entre un fichero del repositorio y un fichero local.
git checkout <nombre fichero>	Volver un fichero local al estado del repositorio del mismo.
Git commit -a	Hacer un commit a todos los archivos modificados.
Git commit -m	Hacer un commit añadiendo un comentario descriptivo.
Git add <nombre fichero>	Añadir un fichero al repositorio.
Git rm <nombre fichero>	Borrar un fichero del repositorio.
Git pull	Actualiza el repositorio local con el repositorio remoto.
Git push	Injecta los cambios realizados en el repositorio local al repositorio remoto.

A continuación vamos a hablar de la tecnología Add-ons SDK.

Como comentamos en el ciclo 1, Add-ons SDK posee dos módulos para desarrollar aplicaciones. En este ciclo se hablará en detalle del API de alto nivel **page-mod**.

Page-mod: Ejecutar secuencias de comandos en el contexto de las páginas web cuya URL coincide con un patrón determinado.

Para utilizar la página-mod, se especifica:

- una o más secuencias de comandos. El SDK llama a estos scripts "guiones de contenido".
- un patrón URL de una página que debe coincidir.

Constructor:

Opciones	Descripción
Include	Introduce un String o un Array de direcciones para que page-mod aplique las definiciones del contenido.
Exclude	Al contrario que include. Estas rutas son descartadas a la hora de aplicar las definiciones de contenido.
contentFileScript	Introduce un String o un Array. Se incluirán en esta página los diferentes archivos javascript al contenido de la página a modificar.
contentScriptWhen	Modifica la manera en la que se carga el contenido javascript. Tres estados: <ul style="list-style-type: none">• start, cargar los contenidos script antes de que el DOM haya sido cargado.• ready, carga los script una vez que el DOM ha sido cargado.• end, carga los script una vez que todos los archivos (DOM, JS, CSS, imágenes) han sido cargados.
attachTo	Adjunta opciones que añadirán la funcionalidad de page-mod a las pestañas. Tres opciones que se pueden combinar: <ul style="list-style-type: none">• exist, page-mod se aplicará automáticamente en las pestañas abiertas.• top, page-mod se aplicará a la pestaña de más alto nivel.• Iframe, page-mod se aplicará al marco flotante dentro de la pestaña.

Eventos:

OnAttach: Este evento se emite cuando el contenido de scripts de la page-mod está asociado a un documento cuya URL coincide con el patrón incluido en page-mod.

La función de escucha se pasa a un objeto trabajador (**worker**), que se puede utilizar para comunicarse con los guiones de contenido que su página-mod ha cargado.

El evento se dispara cada vez que los scripts de los contenidos de la página-mod son llamados por el usuario. Cada vez que se habrá una página que coincida con el patrón URL de page-mod se realizará una instancia nueva del objeto trabajador.

El **Trabajador** puede ser utilizado para la comunicación de script de contenido como se explico en el ciclo 1 en el API de alto nivel Panel.

APIs de terceros

A parte de estas APIs, Add-ons SDK puede usar módulos de terceros que se incluyen dentro de Firefox, pero también módulos propios, y ponerlos a disposición de cualquier persona. Antes de que la herramienta JPM estuviera disponible se hacía muy difícil encontrar módulos desarrollados por otros programadores, instalarlos y actualizarlos.

Con JPM, utilizamos el gestor de paquetes nodo (NGP) que hace envíos a SDK dentro de Firefox. Los desarrolladores pueden publicar sus módulos en NGP para que posteriormente podamos incorporarlos a través de NPM. Ejemplo de instalación de un menú:

- **NPM instalación menuitem –save**

No todo el desarrollo de nuestra aplicación debe estar en nuestro fichero index.js, también podemos separar el desarrollo en módulos propios y exportar los diferentes resultados para que nuestro fichero principal los maneje.

Se pueden crear módulos muy potentes de bajo nivel, pero deben llevar privilegios de “chrome”, lo que nos da acceso a Components object, los cuáles no proporciona restricciones al sistema anfitrión.

Estos componentes objetos son los denominados XPCOM. Son modelos multiplataforma, que permiten ser utilizados por gran cantidad de lenguajes como JavaScript, java, Python, además de C++.

XPCConnect es la herramienta que hace de puente entre JavaScript y XPCOM. Con XPCConnect se pueden usar componentes de código JavaScript e interactuar con los componentes objeto XPCOM. XPCConnect es parte de Firefox y se utiliza activamente en aplicaciones XUL.

Para poder crear estos objetos, como hemos reflejado anteriormente, hay que darle cierta autoridad al sistema anfitrión. Para ello debemos declarar una variable incluyendo los diferentes componentes, pudiendo solamente escribir sus alias:

- Cc (Components.classes)
- Ci (Components.interfaces)
- Cu (Components.utils)
- Cr (Components.results)
- Cm (Components.manager)

Ejemplo:

```
const {Cc,Ci} = require("chrome");
```

Hay tres pasos a seguir a la hora de llamar a un componente XPCOM.

- Obtener el componente.
- Obtener la parte del componente que implementa la interfaz que queremos utilizar.
- Llamar a la función que necesitamos.

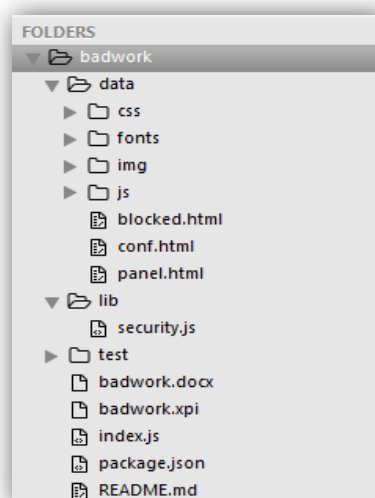
En este proyecto se incluye una librería cuya funcionalidad de control de acceso de la aplicación, a través de una contraseña. Esta librería se llama **security.js**, para ello utilizaré los siguientes XPCOM:

- nsIPromptService: para mostrar una pop-up tipo prompts con diferentes usos.
- nsICryptoHash: para cifrar un string o un fichero con diferentes algoritmos de cifrado.
- nsIScriptableUnicodeConverter: para convertir el dato binario en un dato hexadecimal.

```
const prompts = Cc["@mozilla.org/embedcomp/prompt-service;1"].getService(Ci.nsIPromptService);  
const ch = Cc["@mozilla.org/security/hash;1"].createInstance(Ci.nsICryptoHash);  
const converter = Cc["@mozilla.org/intl/scriptableunicodeconverter"].createInstance(Ci.nsIScriptableUnicodeConverter);
```

** Para más información revisar anexo I.*

En cuanto a la estructura, en la versión actual, he realizado varios cambios para que todos los archivos estén distribuidos de una forma más ordenada. En la imagen siguiente podemos ver como quedaría:



Hay que destacar varias carpetas y ficheros, respecto a la versión anterior:

Data: se han creado subcarpetas para separar los tipos de ficheros dejando en su raíz los HTML:

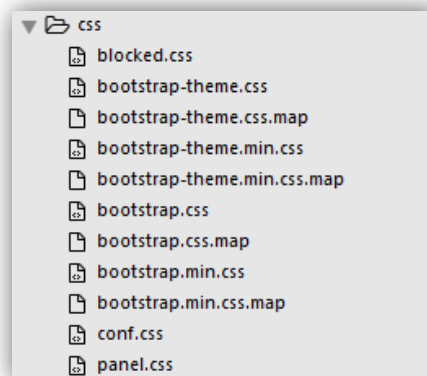
- **Css:** se almacenan las hojas de estilos.
- **Fonts:** las fuentes que se pueden utilizar.
- **Img:** las imágenes de la aplicación.
- **Js:** todos los archivos javascript.

Lib: esta carpeta servirá para almacenar módulos externos, donde creo un fichero security.js que manejará la seguridad de las contraseñas.

En esta versión he incluido librerías de Bootstrap y JQuery para facilitar la programación del diseño y los ficheros javascript.

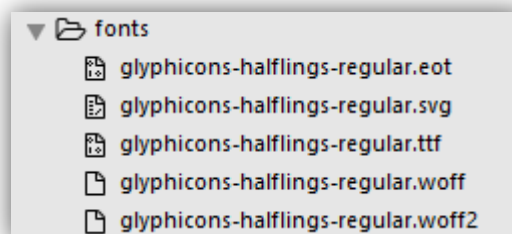
Para incluir la librería he tenido que descargar las versiones más actuales e incluirlas en las diferentes subcarpetas de Data:

La carpeta Css:



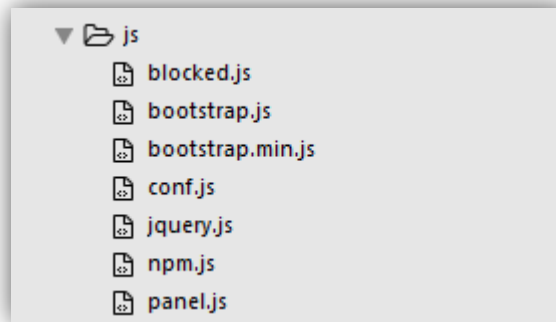
Gran parte de estos archivos Css son incluidos por la librería bootstrap, los más importantes son: bootstrap.css que contiene todas las clases y etiquetas con su estilo característico, y bootstrap.min.css que es la hoja de estilos minimizada.

La carpeta Fonts:



Estos son las fuentes que representan los iconos de bootstrap.

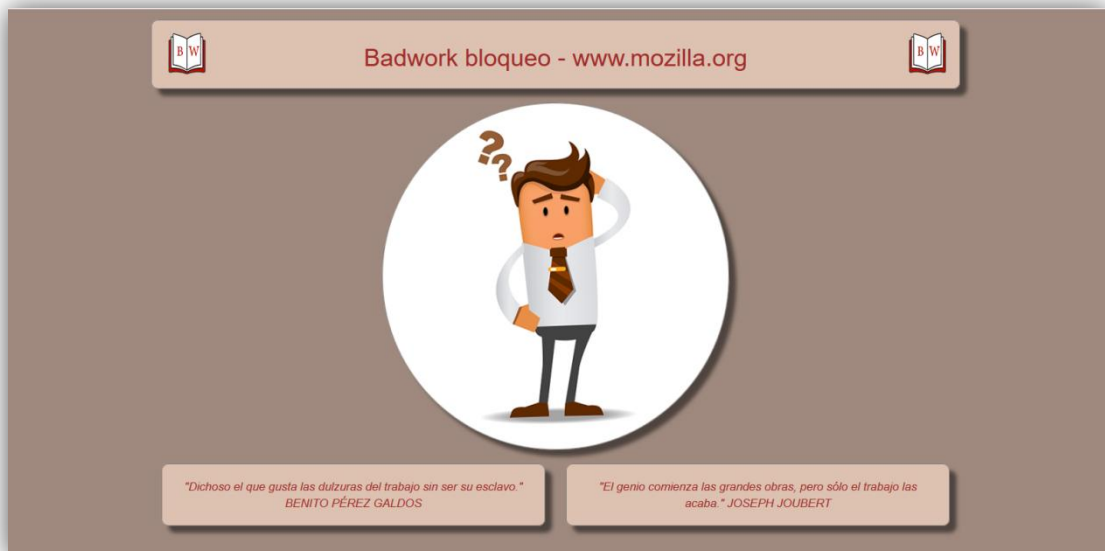
La carpeta JS:



En esta carpeta están incluidos el fichero bootstrap minimizado y la librería jQuery, que es esencial para que funcione bootstrap.

Gracias a bootstrap y jquery se logró el siguiente efecto en las siguientes páginas.

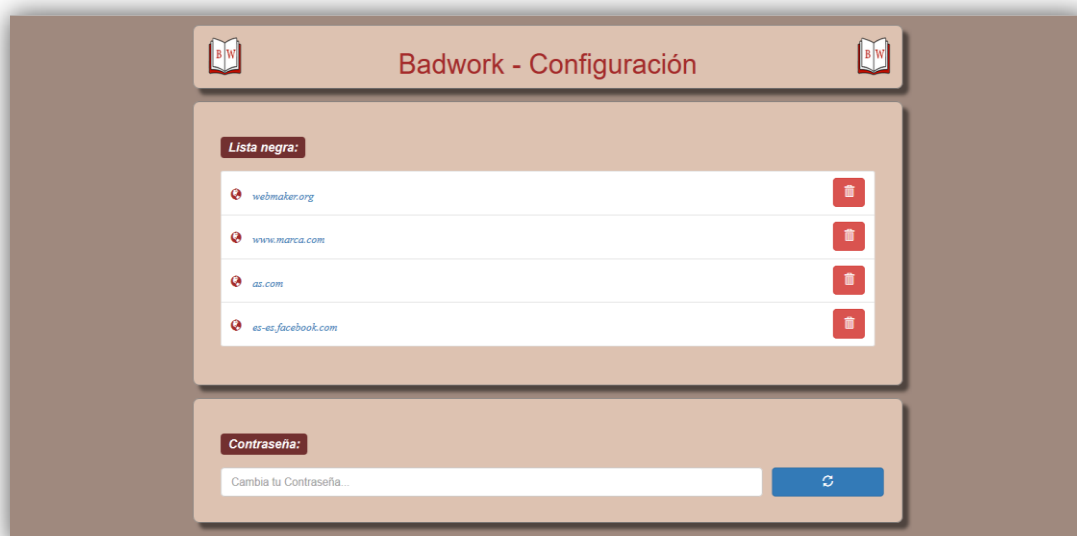
Blocked.html:



Observaciones:

- Se ha creado un header, donde aparecen los iconos de la aplicación, acompañado de un texto explicativo sobre que dominio se ha bloqueado.
- Se ha rediseñado la imagen y se le ha dado una animación CSS.
- Todas las cajas poseen un estilo de sombra, con bordes redondeados.
- En el footer, se han incluido varias citas de ánimo para el buen trabajador, que van cambiando aleatoriamente en cada página bloqueada.

Conf.html:



Observaciones:

- El header es exactamente igual al de BLOCKED.HTML salvo que cambia el texto.
- Se ha creado un contenedor con una lista NEGRA, de las páginas que se han ido bloqueando, y va actualizando según la interacción de la aplicación.
- Se incluyen botones a cada ítem para tener la posibilidad de eliminarlos desde la página de configuración. Se incluye el icono de papelera, icono proporcionado por la librería bootstrap.
- En la parte inferior aparece una zona para cambiar la contraseña. Es un input tipo TEXT con estilo propio al pulsar en él.

A continuación se muestra una tabla con la descripción de las funciones, constructores y mensajes de los archivos `index.js` y `security.js`, que son los ficheros que llevan la lógica de la aplicación:

Index.js	
Elementos	Descripción
Función <code>getCurrentUrl</code>	Devuelve la última url conocida, dependiendo de la pestaña que este activa en ese mismo momento.
Función <code>storagePageBloqued</code>	Se almacena la página que deseo bloquear en la aplicación.
Función <code>deletePage</code>	Borra la página almacenada en la aplicación.
Función <code>checkPages</code>	Verifica si una página está guardada en almacenamiento o no.
Función <code>reloadPageDelete</code>	Recorre todas las pestañas de la ventana del navegador, para recargar las páginas que se han desbloqueado y actualizar la lista negra de la página de configuración.

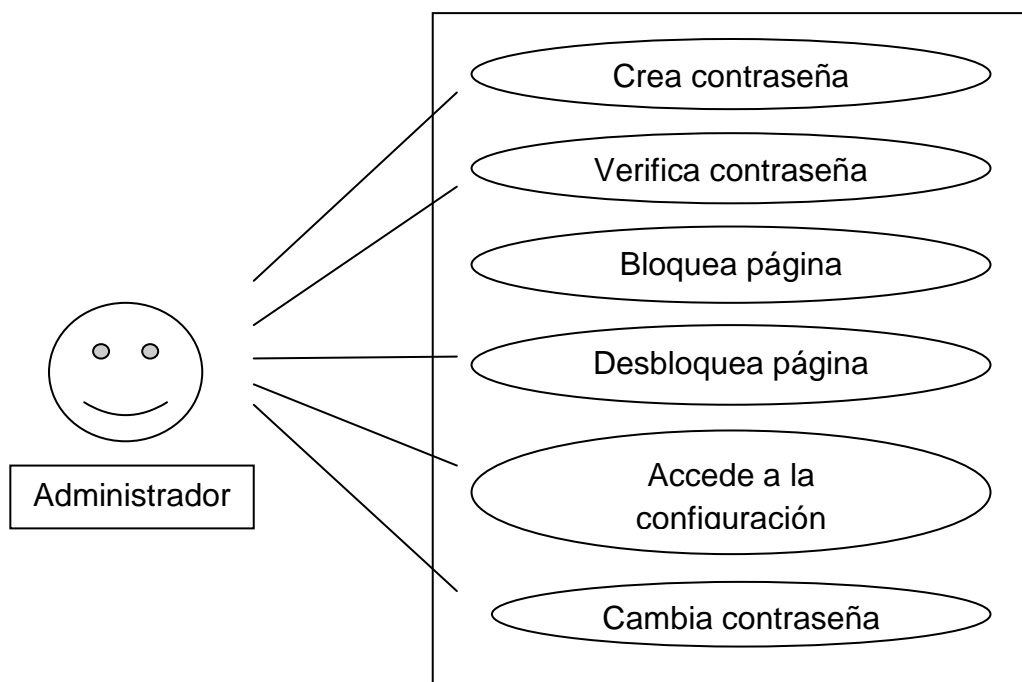
Constructores page-mod	<ul style="list-style-type: none"> • Page-mod para cualquier página. • Page-mod para la página de bloqueo. • Page-mod para la página de configuración.
Mensajes	<ul style="list-style-type: none"> • “storages”: envió de la variable de las paginas almacenadas. • “delete”: indica qué elemento debe ser borrado. • “changePassword”: recibe el password que ha sido cambiado y se cifra en el algoritmo SHA256. • “blocked”: determina la url que debe ser bloqueada. • “unblocked”: determina la url que debe ser desbloqueada. • “config”: determina que se quiere acceder a la configuración de la aplicación.

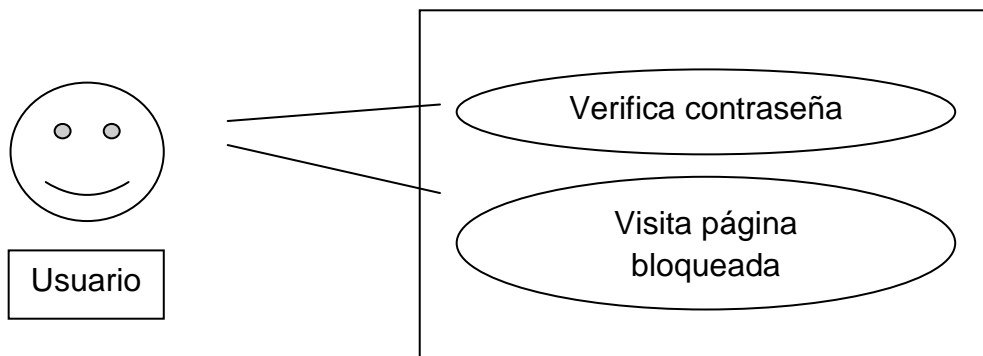
** Para más información revisar anexo II.*

Security.js	
Elementos	Descripción
Función hasPassword	Determina si existe una contraseña almacenada.
Función passwordHash	Codifica la contraseña con un algoritmo de cifrado SHA256.
Función statusPassword	Determina el estado de la contraseña. Recién creada o si está activa.
Función toHexString	Devuelve dos dígitos hexadecimales para cada byte.
Función checkPassword	Verifica si la contraseña introducida es verdadera.

** Para más información revisar anexo III.*

3.2.3.4. Evaluación / Casos de uso y pruebas





Nombre	UC-01 Crear contraseña (administrador).
Secuencia principal	<ol style="list-style-type: none"> 1. El administrador pulsa para iniciar la aplicación. 2. El sistema solicita crear una contraseña. 3. El usuario introduce una contraseña. 4. El sistema recoge y cifra la contraseña en un algoritmo SHA256 y la guarda. 5. El sistema solicita verificar contraseña. 6. El usuario introduce una contraseña. 7. El sistema compara la contraseña guardada con la introducida.
Error / Secuencia alternativa	2.1.i. El sistema tiene ya una contraseña guardada. 2.1.p. Caso de uso finaliza comenzando el UC-02, verificar contraseña.
Post Codificación	El sistema abre el panel de control de acceso.

Nombre	UC-02 Verificar contraseña (usuario).
Secuencia principal	<ol style="list-style-type: none"> 1. El sistema solicita verificar la contraseña. 2. El usuario introduce una contraseña. 3. El sistema recoge la contraseña y la compara con la guardada.
Error / Secuencia alternativa	3.1.i. El usuario ingresa una contraseña correcta. 3.1.p. El sistema abre el panel de control de acceso.
Post Codificación	El sistema muestra un mensaje de alerta de contraseña incorrecta.

Nombre	UC-03 Bloquear página.
Secuencia principal	<ol style="list-style-type: none"> 1. El administrador solicita bloquear www.mozilla.org. 2. El sistema recoge la información de la url. 3. El sistema almacena el dominio en un array. 4. El sistema redirecciona a la página de bloqueo.
Error / Secuencia alternativa	<ol style="list-style-type: none"> 1.1.i. El administrador solicita www.mozilla.org en una nueva pestaña. 1.1.p La página está bloqueada. 1.2.i. El administrador abre varias pestañas con la url www.mozilla.org y solicita en una pestaña el bloqueo de la página. 1.2.p. Las pestañas que coinciden con esa url son redireccionadas a la página de bloqueo. 1.2.i. El usuario solicita desbloquear www.mozilla.org. Caso de uso finaliza y comienza UC-04 desde la secuencia 2.
Post Codificación	El usuario ve que la página está bloqueada.

Nombre	UC-04 Desbloquear página.
Secuencia principal	<ol style="list-style-type: none"> 1. El administrador solicita desbloquear www.mozilla.org. 2. El sistema encuentra la página para desbloquear. 3. El sistema borra el dominio del array. 4. El sistema redirecciona a la página www.mozilla.org.
Error / Secuencia alternativa	<ol style="list-style-type: none"> 1.1.i. El usuario abre varias pestañas con la url www.mozilla.org, que está bloqueada, y solicita desbloquear en una de ellas. 1.1.p. Todas las pestañas donde coincida la url serán desbloqueadas.
Post Codificación	El usuario vuelve a ver la página.

Nombre	UC-05 Cambiar contraseña.
Secuencia principal	<ol style="list-style-type: none"> 1. El administrador introduce una nueva contraseña. 2. El administrador pulsa el botón cambiar contraseña. 3. El sistema sustituye la contraseña antigua por la nueva 4. El sistema alerta que la contraseña ha sido cambiada. 5. El sistema reinicia la solicitud de verificación. 6. El sistema cierra la pestaña de configuración.
Error / Secuencia alternativa	<ol style="list-style-type: none"> 1.2.i. El administrador introduce una contraseña poco segura. 1.2.p. ERROR. El sistema no avisa de que la contraseña no es segura y realiza el cambio de contraseña sin verificación. 2.1.i. El administrador pulsa botón sin introducir contraseña. 2.1.p. No se realiza el cambio de contraseña. ERROR no se avisa al usuario de que debe ingresar una contraseña.
Post Codificación	La contraseña de acceso al panel de control cambió.

3.2.3.5. Construcción y entrega

En el ciclo 1, para que Mozilla firmara nuestra aplicación utilizamos JPM, con la particularidad de que la aplicación no estaría en la lista de complemento de Mozilla, sino que nos tocaría a los desarrolladores distribuirla por cuenta propia.

En este ciclo vamos a firmar el complemento a través de la página de add-ons de Mozilla, para que esta vez aparezca en la lista de extensiones y toda la comunidad pueda descargársela e instalarla.

Requisitos:

- generar un archivo XPI de la aplicación (**JPM XPI**).
- Ingresar en <https://addons.mozilla.org>
- Subir el complemento.

Pasos a seguir:

Envía tu complemento. En este proceso subiremos nuestro archivo XPI, y una vez subido se irán haciendo una serie de validaciones del complemento.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar
2. **Envía tu complemento**
3. Describe tu complemento
4. Agregar imágenes
5. Seleccionar una licencia
6. Seleccionar un proceso de revisión
7. ¡Ya está!

Paso 2. Envía tu complemento

Use los siguientes campos para enviar tu paquete y seleccionar cualquier restricción en la plataforma. Tras enviarlo, una serie de pruebas de validación automática se ejecutarán sobre tu fichero.

¿Quieres que se distribuya tu complemento en esta página?

☐ No listar mi complemento en este sitio ?

Selecciona un archivo...

Your add-on should end with .zip, .xpi, .jar or .xul






¿Con qué plataformas es compatible este archivo?

☒ Todas las plataformas ☐ Linux ☐ Mac OS X ☐ Windows ☐ Android

Continuar

Resultados de la validación para 87944e776b214a5590371b4525d931c9_badwork.xpi

Validado el: June 2, 2016

				
Pruebas generales	Pruebas de seguridad	Pruebas de extensiones	Pruebas de localización	Pruebas de compatibilidad
0 errores, 0 avisos, 0 mensajes	0 errores, 0 avisos, 0 mensajes	0 errores, 28 avisos, 0 mensajes	0 errores, 0 avisos, 0 mensajes	0 errores, 0 avisos, 0 mensajes

El complemento ha pasado la validación.

Si el proceso se realizó con éxito podemos proseguir.

Describe tu complemento. En este apartado se debe hacer un resumen para saber en qué consiste tu aplicación, y elegir una categoría para clasificarla a la hora de buscar un complemento.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar
2. Envía tu complemento
3. **Describe tu complemento**
4. Agregar imágenes
5. Seleccionar una licencia
6. Seleccionar un proceso de revisión
7. ¡Ya está!

Paso 3. Describir

Nombre y versión: badwork

1.0.1

La página de detalles será: <https://addons.mozilla.org/.../badwork> Editar

Escribe un pequeño resumen:

Control de acceso a páginas no deseadas por el administrador. Tendrás a tus empleados controlados con la facilidad de pulsar un botón. El administrador poseerá una contraseña para tener acceso a la administración de las páginas.

Este resumen se mostrará en listas y búsquedas.

22 caracteres restantes.

Selecciona hasta 2 categorías de Firefox para este complemento:

- | | |
|--------------------------------------------------------------------------------|------------------------------------------------------------|
| <input type="checkbox"/> Administración de descargas | <input type="checkbox"/> Herramientas de búsqueda |
| <input type="checkbox"/> Alertas y actualizaciones | <input type="checkbox"/> Juegos y ocio |
| <input type="checkbox"/> Apariencia | <input type="checkbox"/> Marcadores |
| <input type="checkbox"/> Asistencia de idiomas | <input type="checkbox"/> Pestañas |
| <input type="checkbox"/> Compras | <input checked="" type="checkbox"/> Privacidad y seguridad |
| <input type="checkbox"/> Desarrollo web | <input type="checkbox"/> RSS, noticias y blogs |
| <input type="checkbox"/> Fotos, música y vídeos | <input type="checkbox"/> Redes sociales y comunicación |
| <input type="checkbox"/> Mi complemento no encaja en ninguna de las categorías | |

Selecciona hasta 2 categorías de Firefox para Android para este complemento:

- | | |
|----------------------------------------------------|---------------------------------------------------------------|
| <input type="checkbox"/> Canales, noticias y blogs | <input type="checkbox"/> Interfaz de usuario |
| <input type="checkbox"/> Compras | <input type="checkbox"/> Opciones del dispositivo y ubicación |
| <input type="checkbox"/> Deportes y juegos | <input type="checkbox"/> Redes sociales |
| <input type="checkbox"/> Experimentales | <input type="checkbox"/> Rendimiento |
| <input type="checkbox"/> Fotos y multimedia | <input checked="" type="checkbox"/> Seguridad y privacidad |

Agregar imágenes. Agregamos las imágenes que van a representar los iconos de la aplicación y el funcionamiento de la misma.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar

2. Envía tu complemento

3. Describe tu complemento

4. Agregar imágenes

5. Seleccionar una licencia


6. Seleccionar un proceso de revisión


7. ¡Ya está!


Paso 4. Agregar imágenes

Los iconos y las capturas de pantalla personalizadas llaman la atención y ayudan a los usuarios a entender lo que hace. Recomendamos enormemente que subas un icono personalizado ya que, en algunas partes de la página web, solo aparecerán el icono y el nombre.

Selecciona un icono para tu complemento:





32x32px


64x64px


Subir un icono personalizado...

Se admite PNG y JPG. Los iconos se redimensionarán a 64 x 64 pixels si son mayores.

Proporciona al menos una captura de pantalla de tu complemento:



Proporciona un título para esta captura de pantalla:
Password



Proporciona un título para esta captura de pantalla:
Mandos de control

Elegir una licencia: La licencia MIT pone muy pocas restricciones en la reutilización de software. Los derechos otorgados al Usuario final en la licencia MIT, incluyen: el derecho a usar, copiar, modificar, fusionar, publicar, distribuir, cambiar la licencia y/o vender el software.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar

2. Envía tu complemento

3. Describe tu complemento

4. Agregar imágenes

5. Seleccionar una licencia

6. Seleccionar un proceso de revisión

7. ¡Ya está!

Paso 5. Elige una licencia

Es obligatorio para todos los complementos indicar los términos bajo los que está licenciado su código fuente. Selecciona una licencia de la siguiente lista o introduce tu propia licencia.

Selecciona una licencia para tu complemento:

☐ Mozilla Public License, version 2.0 [Detalles](#)

☐ Licencia pública GNU, versión 2.0 [Detalles](#)

☐ Licencia pública GNU, versión 3.0 [Detalles](#)

☐ Licencia pública menor GNU Lesser, versión 2.1 [Detalles](#)

☐ Licencia pública menor GNU Lesser, versión 3.0 [Detalles](#)

☒ Licencia MIT/X11 [Detalles](#)

☐ Licencia BSD [Detalles](#)

☐ Otro

☐ Este complemento tiene una Licencia de Usuario Final:

☐ Este complemento tiene una política de privacidad:

Continuar

31

Seleccionar un proceso de revisión: en este apartado debemos elegir el método de revisión. Esto puede tardar varios días dependiendo del método que elijamos.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar
2. Envía tu complemento
3. Describe tu complemento
4. Agregar imágenes
5. Seleccionar una licencia
- 6. Seleccionar un proceso de revisión**
7. ¡Ya está!

Paso 6. Elige un proceso de revisión

Todos los complementos alojados en nuestra galería deben ser revisados por un editor antes de aparecer en las categorías o resultados de búsqueda. Mientras espera por revisión, tu complemento puede todavía ser accesible a través de la URL directa. Te sugerimos que elijas el proceso de revisión que mejor se adapte a tu complemento.

Revisión completa

Una revisión completa del código y funcionalidad de tu complemento. [Más información...](#)

- Apropiado para complementos pulidos
- Requerido si el complemento está incluido en un instalador
- La revisión debe tener lugar en 10 días
- Revisión de versiones siguientes en 5 días
- Libre de advertencias de instalación y sin limitaciones de características

Elegir revisión completa

Revisión preliminar

Una revisión más rápida del código de tu complemento para cualquier problema importante. [Más información...](#)

- Apropiado para complementos experimentales
- La valoración debe tener lugar en tres días
- Algunas limitaciones de características
- Binarios y complementos ofuscados no legibles

Elegir revisión preliminar

¡Ya está! El proceso se completo correctamente. Ya esta nuestra versión liberada, pero no revisada, esto quiere decir que hasta que el desarrollador no reciba un mail con los resultados de la revisión, no aparecerá en la búsqueda de galería de complementos de Mozilla, pero si a través de la url proporcionada.

Envía un nuevo complemento

PROCESO DE ENVÍO

1. Comenzar
2. Envía tu complemento
3. Describe tu complemento
4. Agregar imágenes
5. Seleccionar una licencia
6. Seleccionar un proceso de revisión
- 7. ¡Ya está!**

¡Ya está!

Tu complemento ha sido enviado a la cola de revisión preliminar.

You'll receive an email once it has been reviewed by an editor and signed. Once it has been signed you will be able to install it:

<https://addons.mozilla.org/es/firefox/addon/badwork/>

Siguientes pasos:

- [Edita su lista](#) para dar más detalles.
- Explica a tus usuarios por qué lo has creado en tu [Perfil de desarrollador](#).
- Visita y suscríbete a tu [monitor de actividad](#) de tu complemento para estar informado sobre valoraciones, colecciones y mucho más.
- Ver los [tiempos de espera](#) aproximados de la cola de revisión.

¡Adelántate en la Lista de revisión!

Conviértete en revisor de AMO hoy mismo y tus complementos tendrán preferencia en las revisiones.

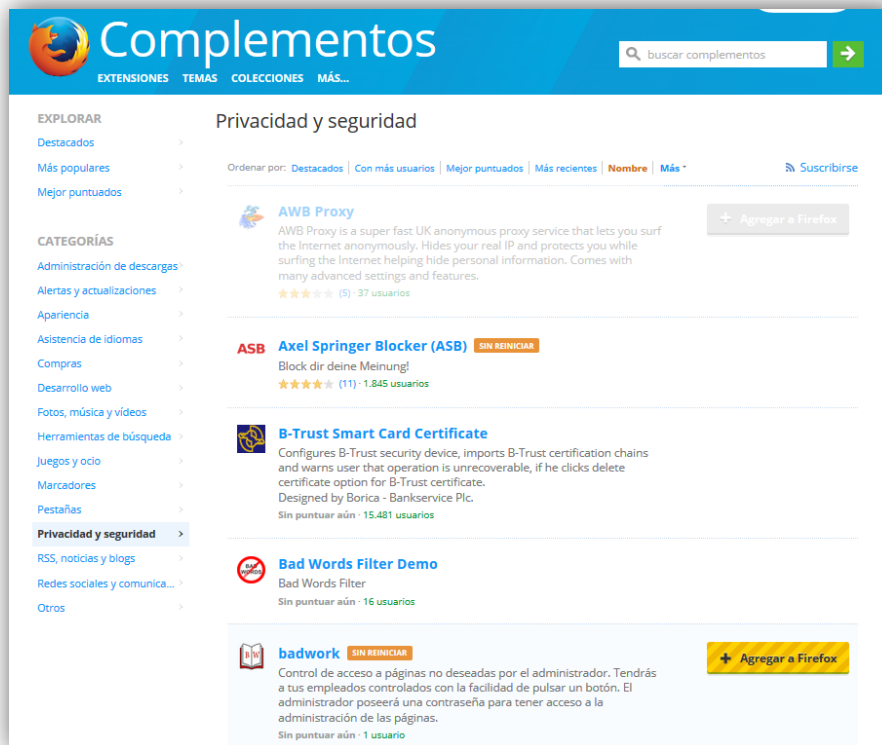
Más información

Complemento no revisado:

The screenshot shows the Mozilla Add-ons page for the 'badwork' extension. The page has a blue header with the Mozilla logo and navigation links: 'karlos', 'Herramientas', 'Otras aplicaciones', and 'mozilla'. A search bar with the text 'buscar complementos' is on the right. Below the header, the extension 'badwork' by 'karlos' is displayed. It has a version of 1.0.1 and a 'SIN REINICIAR' (Not Restarted) status. The description states: 'Control de acceso a páginas no deseadas por el administrador. Tendrás a tus empleados controlados con la facilidad de pulsar un botón. El administrador poseerá una contraseña para tener acceso a la administración de las páginas.' A yellow button 'Agregar a Firefox' is visible. To the right, it says 'Sin puntuar aún' and '0 valoraciones de los usuarios', with a 'Gestionar' button. Below the extension details, there are three preview images showing the extension's interface. At the bottom, the 'Acerca de este complemento' section states: 'Esta aplicación esta orientada a un entorno laboral, donde el trabajador tiene que ser consciente que tiene que trabajar y no distraerse. La aplicación empezará pidiendo credenciales, si son correctas, el usuario tendrá acceso a la parte administrativa de la...'. On the right, it lists 'Versión 1.0.1', 'Última actualización: June 2, 2016', and 'Publicado bajo Licencia MIT/X11'.

Complemento revisado:

The screenshot shows the Mozilla Add-ons page for the 'badwork' extension, now marked as 'Reviewed'. The layout is similar to the previous one, but the status is 'REVISADO' (Reviewed) instead of 'SIN REINICIAR'. The description remains the same. The 'Agregar a Firefox' button is still present. The 'Acerca de este complemento' section at the bottom is identical to the previous one. The right side of the page shows 'Sin puntuar aún' and '0 valoraciones de los usuarios', but now it also lists '1 usuario'. The version and update information remain the same: 'Versión 1.0.1', 'Última actualización: June 2, 2016', and 'Publicado bajo Licencia MIT/X11'.



4. Resultado y discusión

Al empezar a realizar el proyecto y tras documentarme, decido utilizar la tecnología “XUL”. Cuando comienzo a programar con esta tecnología me encuentro diversas dificultades:

- Cuando trabajo localmente con los archivos con extensión “.xul” hayo incompatibilidades con las versiones más actuales de Firefox.
- A la hora de instalar el API (Application Programming Interface) necesita la firma de verificación del grupo de desarrollo de Mozilla.

Por estos inconvenientes me decido finalmente a realizar el proyecto en una tecnología que no esté tan desfasada e implementarlo utilizando Add-on SDK (Kit de Desarrollo Software), que te permite desarrollar APIs de Firefox siguiendo las tecnologías Web estándar: JavaScript, Css y Html.

Add-ons SDK tiene numerosas ventajas respecto a XUL:

- Posee una herramienta por línea de comando denominada **JPM**, que tiene diferentes funciones:
 1. Inicializar un proyecto desde cero (creando una estructura base de directorios y archivos). **Jpm init**
 2. Ejecutar el proyecto en local. **Jpm run**
 3. Probar.

4. Empaquetar proyecto para posteriormente instalarlo. **Jpm xpi**

- JPM está basada en Node.js y sustituye a la herramienta CFX.
- JPM se puede utilizar en versiones de Firefox 38 en adelante.
- Posee módulos ya definidos que te ayudan a la hora de crear el API e interactuar con el usuario.
- Puedes añadir módulos de terceros o incluso poner tus creaciones.

Se ha estudiado incluir Git en sublime, pero se desestimó este proceso ya que en mi opinión es menos engorroso escribir los comandos de git en la terminal. Sin embargo, en sublime hay que dar o varios botones o poner un comando para después insertar otro.

A parte, utilizar la terminal hace que seamos más profesionales y practiquemos los diferentes comandos y no los olvidemos.

5. Conclusiones

Estos casi tres meses desarrollando este proyecto me han aportado grandes cosas. Una de ellas ha sido el saber mis habilidades y limitaciones a la hora de utilizar y programar una tecnología desde cero. Ha sido un trabajo de muchas horas de documentación y tutoriales antes de ponerme a programar la codificación. Gracias a este proyecto he mejorado mis técnicas y comprensión en el lenguaje JavaScript, que pienso que es el futuro, debido a que quita mucho peso de trabajo a los servidores.

En mi opinión Add-ons SDK es una tecnología sencilla si quieres introducirte en la programación de complementos para Mozilla. Posee una documentación, en inglés, fácil de entender y con muchos ejemplos. Por otra parte, la propia comunidad de Mozilla te puede ayudar en foros y eso supone una gran satisfacción y te anima a que tú también contribuyas con otros programadores.

Aunque esta aplicación es básica, creo que tiene grandes posibilidades y seguiré desarrollándola para darle una versión más estable y profesional.

Por falta de tiempo hubo versiones que no se pudieron realizar, como por ejemplo:

- Páginas almacenadas por defecto para su bloqueo por categorías: Redes Sociales, Noticia Deportiva, Juegos. Pudiendo activarse y desactivarse.
- Botón para importar una lista de páginas web prefijadas que se desean bloquear.
- Modificación de roles de la aplicación, dependiendo para quién fuera orientado, dando a un botón cambiaría todo el estilo y las imágenes. Por ejemplo para un entorno escolar, familiar o temática variada.

Como he mencionado anteriormente seguiré desarrollando esta aplicación para llevarla a un nivel más alto, pero si tuviera que desarrollar otra aplicación distinta para Mozilla creo que utilizaría Web extensión, que es una tecnología más actual, ya que se está intentando alejarse de Add-on SDK, aunque lentamente.

6. Bibliografía y referencias

http://soporte.eset-la.com/kb2798/?locale=es_ES

https://es.wikipedia.org/wiki/Desarrollo_en_espiral

http://www.sites.upiicsa.ipn.mx/polilibros/portal/polilibros/P_externos/Administracion_informatica_de_las_organizaciones_Ramon_E_Enriquez_Gonzalez/AIO2_Mod_ESPIRAL.html

<http://software1nathalygrijalva.blogspot.com.es/2012/10/modelo-espiral.html>

<http://developer.mozilla.org/en-US/docs/Mozilla/Tech/XUL/Tutorial>

<https://developer.mozilla.org/en-US/Add-ons/SDK>

https://es.wikipedia.org/wiki/Kit_de_desarrollo_de_software

<https://developer.mozilla.org/en-US/Add-ons/SDK/Tools/jpm#Installation>

<https://nodejs.org/en/>

<https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XPCOM/Reference/Interface>

<https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XPCOM/Reference/Interface/nsIScriptableUnicodeConverter>

<https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XPCOM/Reference/Interface/nsICryptoHash>

<https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XPCOM/Reference/Interface/nsIPromptService>

Imágenes:

Trabajador con dudas fue facilitada por Vector desde:

<http://www.freepik.es/fotos-vectores-gratis/negocios>

Iconos (libro y engranajes) facilitada por Freepik desde:

<http://www.flaticon.com>

7. Anexos/Otros

7.1. Anexo I

NsIPromptService: esta interfaz sirve para mostrar diálogos simples.

Métodos	Descripción
Alert	Muestra un mensaje de alerta con un botón. Trabaja de la misma manera que Windows.alert, con la diferencia que acepta poner un título a la ventana modal.
promptPassword	Muestra un diálogo con campo editable tipo password y un checkbox opcional.

NsICryptoHash: esta interfaz es usada para cifrar ficheros o datos con varios algoritmos de cifrado (MD2, MD5, SHA256, SHA384 y SHA512).

Métodos	Descripción
Init	Inicia el objeto de cifrado.
Update	Añade una matriz de datos que deben ser cifrados por el objeto inicializado.
Finish	Finaliza el objeto de cifrado iniciado, completándolo con los datos cifrados actuales.

NsIScriptableUnicodeConverter: esta interfaz convierte cadenas utilizadas en un script, en datos binarios para utilizarlos a la hora de cifrar.

Métodos	Descripción
convertToByteArray	Convierte un cadena en un array de bytes.

7.2. Anexo II

Funciones:

GetCurrentUrl:

```
// dar la ultima url conocida
function getCurrentUrl(){
    if(tabs.activeTab.url == data.url("blocked.html")){
        tabs.on('activate', function(tab) {
            url = tab.title;
            panel.port.emit("load", url);
        });
    } else {
        url = urls.URL(tabs.activeTab.url).host;
        panel.port.emit("load", url);
    }
    panel.port.on("upload", function(page){
        url = page;
    });
    return url;
}
```

StoragePagesBlocked:

```
// almacenar bloqueo
function storagePageBlocked(tab) {
    var exist = false;
    for each (page in ss.storage.pages) {
        if(page == tab) {
            exist = true;
        }
    }
    if (!exist) {
        ss.storage.pages.push(tab);
    }
    var newStorage = ss.storage.pages;
    return newStorage;
}
```

DeletePage:

```
// borrar pagina del almacen
function deletePage(tab) {
    for (var i = 0; i < ss.storage.pages.length; i++) {
        if(tab == ss.storage.pages[i]) {
            ss.storage.pages.splice(i,1);
        }
    }
}
```

CheckedPage:

```
// chequear la pagina
function checkedPage(tab){
    var blocked = false;
    for each (page in ss.storage.pages) {
        if (page == tab) {
            blocked = true;
        }
    }
    return blocked;
}
```

ReloadPageDelete:

```
// recorrer paginas para recargar las desbloqueadas de la configuracion
function reloadPageDelete(page) {

    for (let tab of tabs) {
        title = tab.title;
        // redireccionar paginas para desbloquearlas
        if(page == title) {
            tab.url = "http://" + page;
        }
        // recargar pagina configuracion
        if(conf == title) {
            tab.reload();
        }
    }
}
```

Constructores de page-mod para diferentes páginas:

Cualquier página:

```
// cualquier pagina
pageMod.PageMod({
    include: "*",
    exclude:[data.url("blocked.html"),data.url("config.html")],
    contentScriptWhen : "ready",
    attachTo : [ "existing" , "top" ] ,
    onAttach: function onAttach(worker) {
        lastDomain = getCurrentUrl();
        if(checkedPage(lastDomain)) {
            worker.tab.url = data.url("blocked.html");
        }
    }
});
```

Este constructor va a atacar a todas las páginas de una sola vez, excluyendo a las páginas cuya URL sea BLOCKED.HTML y CONFIG.HTML. A este constructor se le asigna un evento para comprobar si la página debe estar bloqueada, si fuera cierto, redirige a BLOCKED.HTML.

Para la página de bloqueo:

```
// pagina de bloqueo
pageMod.PageMod({
  include: data.url("blocked.html"),
  contentScriptWhen: "end",
  contentScriptFile: [data.url("js/jquery.js"), data.url("js/bootstrap.min.js"), data.url("js/blocked.js")],
  onAttach: function onAttach(worker) {
    worker.tab.title = lastDomain;
    worker.port.emit("domainPage", worker.tab.title);
  }
});
```

Este constructor sólo atacará a las páginas cuya URL sean Blocked.HTML, se incluirán varios archivos JavaScript (BootStrap, JQuery y blocked.js).

Se asignará una función que ataque a la página activa, modificando el título de la misma y enviado un mensaje, que será recibido por el archivo blocked.js para poder modificar el contenido de la página.

Archivo Blocked.js incrustado en la página:

```
$("#article").hide();
var domain = $("#dominio");
var domainBlocked = "";
var p = $("p").length;
var a = Math.floor((Math.random() * 5) + 1);
var b = a - 1;
if(b == 0) {
  b = p;
}
$("#frase-animo" + a).show();
$("#frase-animo" + b).show();
self.port.on("domainPage", function(tab) {
  domain.html(tab);
  domainBlocked = domain.html();
});
```

En este fichero js ocultamos las frases de ánimo y calculamos un número aleatorio para que sólo salgan dos cada vez. Se recibe la variable enviada por index.js para modificar el header de la página.

Para la página de configuración:

```
// pagina de configuracion
pageMod.PageMod({
  include: data.url("conf.html"),
  contentScriptWhen: "end",
  contentScriptFile: [data.url("js/jquery.js"), data.url("js/bootstrap.min.js"), data.url("js/conf.js")],
  onAttach: function onAttach(worker) {
    worker.tab.title = "config";
    worker.port.emit("storages", ss.storage.pages);
    // mensaje de borrado de lista
    worker.port.on("delete", function(index){
      var pageToDelete = ss.storage.pages[index];
      ss.storage.pages.splice(index,1);
      reloadPageDelete(pageToDelete);
    });
    // mensaje de cambio de contraseña
    worker.port.on("changePassword", function(password){
      var newPassword = security.passwordHash(password);
      ss.storage.password = newPassword;
      isActive = false;
      worker.tab.close();
    });
  }
});
```

Constructor muy parecido a la página de bloqueo, salvo por los diferentes mensajes que son enviados y recibidos.

- Mensajes enviados de index.js a conf.js:
 - “storages”: envío de la variable de las páginas almacenadas.
- Mensajes recibidos de index.js desde conf.js:
 - “delete”: recibe un índice (numérico), esto indica el elemento que debe ser borrado del array almacenado.
 - “changePassword”: recibe el password que ha sido cambiado y se cifra en el algoritmo SHA256. Se reinicia la petición de la contraseña al realizar el cambio.
- **Archivo Conf.js incrustado en la página:**
 - Recibe el array con las páginas bloqueadas, según el número de páginas se irán creando diferentes etiquetas HTML en el DOM.
 - Las etiquetas serán padres de la etiqueta <A> y <BUTTON>.
 - La etiqueta <A> contendrá el texto de cada ítem del array.
 - <BUTTON> tendrá como identificador el índice representativo del ítem del array.
 - A cada <BUTTON> se le asignará un evento “onclick”, que enviará al index.js información sobre que página hay que borrar del bloqueo.
 - Por último en el apartado de password, se asignará también un evento “onclick”, recogiendo el nuevo password introducido en el input para posteriormente enviarlo al fichero index.js.

```

// recibir el array con las paginas almacenadas
self.port.on("storages", function(array){
    contenedorLista = $("#lista");
    for (var i = 1; i < array.length; i++) {
        var li = $('<li/>', {
            'class' : 'list-group-item',
        });
        var span = $('<span/>', {
            html : "<i><a href='http://'+array[i]+'> " + array[i] + "</a></i>",
            'class' : 'pagina glyphicon glyphicon-globe'
        });
        var btn = $('<button/>', {
            'id': i,
            html: '<span class="glyphicon glyphicon-trash"></span>',
            'class': 'btn btn-danger fl-drch btn-lg'
        });
        li.append(span);
        li.append(btn);
        contenedorLista.append(li);
        $("##" + i).click(function(event){
            var index = $(this).attr("id")
            self.port.emit("delete", index);
        });
    }
});
// enviar contraseña
$("#btn-cambiar-password").click(function(event){
    var newPassword = $("#password").val();
    if(newPassword != "") {
        window.alert("Contraseña cambiada");
        self.port.emit("changePassword", newPassword);
    }
});

```

7.3. Anexo III

Funciones:

HasPassword:

```
// determina si existe password
function hasPassword() {
    return ss.storage.password ? true : false;
}
```

PasswordHash:

```
// encripta el password
function passwordHash(password) {
    converter.charset = "UTF-8";
    var arr = {};
    var data = converter.convertToByteArray(password, arr);
    ch.init(ch.SHA256);
    ch.update(data, data.length);
    var hash = ch.finish(false);
    // convertir el dato binario en un dato hexadecimal
    var stringHash = Array.from(hash, (c, i) => toHexString(hash.charCodeAt(i))).join("");
    return stringHash;
}
```

StatusPassword:

```
// determina estado del password
function statusPassword() {
    if(ss.storage.password) {
        return PASSWORD_CREATED;
    } else {
        var input = {value: null};
        var check = {value: true};
        var admin = prompts.promptPassword(null, "BadWork - Admin", "Crea la clave de acceso:", input, "Guardar:", check);
        if (admin) {
            ss.storage.password = passwordHash(input.value);
        }
        return PASSWORD_ACTIVE;
    }
}
```

ToHexString:

```
// devuelve dos digitos hexadecimales para cada byte
function toHexString(charCode) {
    return ("0" + charCode.toString(16)).slice(-2);
}
```

CheckPassword:

```
// chequea el password
function checkPassword() {
  if(check) {
    return true;
  } else {
    var input = {value: null};
    var check = {value: true};
    var password = prompts.promptPassword(null, "BadWork - Security", "Verifica la clave de acceso:", input, "Guardar:", check);
    if(password) {
      if(passwordHash(input.value) == ss.storage.password) {
        return true;
      } else {
        prompts.alert(null, "BadWork - ERROR", "contraseña incorrecta.");
        return false;
      }
    }
    return false;
  }
}
```