DWEC_P04: Exercicis de la U. D. 4.

Els exercicis que estan en ver són d'entrega voluntària per al grup de DUAL.

4.1. Demanar el nom a l'usuari i mostrar-lo en un paràgraf, en majúscula en la barra d'estat, en un alert i en majúscula en el títol de la pàgina principal. Què ha passat amb la barra d'estat? Aquest exercici toca:

- Funcions javascript.
- Event onload() en body.
- window.prompt del BOM.
- window.alert() del BOM.
- window.status del BOM.
- document.title.
- getElementById() del DOM.
- Propietat innerHTML.

4.2. Mostra la hora en la pàgina i que vagi canviant cada segon. Aquest exercici toca:

- Objecte Date de JS i els seus mètodes.
- getElementById() del DOM.
- propietat innerHTML.
- window.setInterval() del BOM.

4.3. Fer una pàgina que tengui un enllaç per anar cap enrere en l'historial i un altre per anar cap envant. Òbviament, s'ha d'implementar fent servir JS. Aquest exercici toca:

- window.history.back();
- window.history.forward();

4.4. Crea una pàgina on aparegui la següent informació (Resultats d'exemple):

- **Altura ventana:** 1043
- **Anchura ventana:** 1920
- **Profunidad color:** 24 bits
- **Resolución atura:** 1080
- **Resolución anchura:** 1920
- **Navegador:** Mozilla
- **Navegador2:** Netscape
- **Version:** 5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.130 Safari/537.36
- **Cookies:** true
- **Java:** true
- **userAgent:** Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.130 Safari/537.36
- **Idioma Navegador:** es
- **Idioma Navegador2:** undefined
- **Idioma SO:** undefined
- **Idioma preferido usuario:** undefined
- **Sistema operativo:** Linux x86_64
- **Tipos mime Navegador 0 :**

- **Tipos mime Navegador 1 :** Shockwave Flash
- **Tipos mime Navegador 2 :** FutureSplash Player
- **Tipos mime Navegador 3 :**
- **Tipos mime Navegador 4 :** Widevine Content Decryption Module
- **Tipos mime Navegador 5 :** Native Client Executable
- **Tipos mime Navegador 6 :** Portable Native Client Executable
- **Tipos mime Navegador 7 :** Portable Document Format
-
- **Plugins Navegador 0:** Chrome PDF Viewer, mhjfbmdgcfjbbpaeojofohoefgiehjai, , undefined
- **Plugins Navegador 1:** Shockwave Flash, libpepflashplayer.so, Shockwave Flash 18.0 r0, undefined
- **Plugins Navegador 2:** Chrome Remote Desktop Viewer, internal-remoting-viewer, This plugin allows you to securely access other computers that have been shared with you. To use this plugin you must first install the Chrome Remote Desktop webapp., undefined
- **Plugins Navegador 3:** Widevine Content Decryption Module, libwidevinecdmadapter.so, Enables Widevine licenses for playback of HTML audio/video content. (version: 1.4.8.823), undefined
- **Plugins Navegador 4:** Native Client, internal-nacl-plugin, , undefined
- **Plugins Navegador 5:** Chrome PDF Viewer, internal-pdf-viewer, Portable Document Format, undefined

NOTA: La part de tipus MIME i de plugins s'han de fer amb un bucle. Aquest exercici toca:

- screen.availHeight;
- screen.availWidth;
- screen.colorDepth;
- screen.height;
- screen.width;
- navigator.appCodeName;
- navigator.appName;
- navigator.appVersion;
- navigator.cookieEnabled;
- navigator.javaEnabled();
- navigator.userAgent;
- window.navigator.language;
- window.navigator.languajeBrowser;
- window.navigator.mimeTypes;
- window.navigator.platform;
- window.navigator.plugins;
- window.navigator.systemLanguage;
- window.navigator.userLanguage;

4.5. Explica el mètode window.open. Posa 3 exemples.

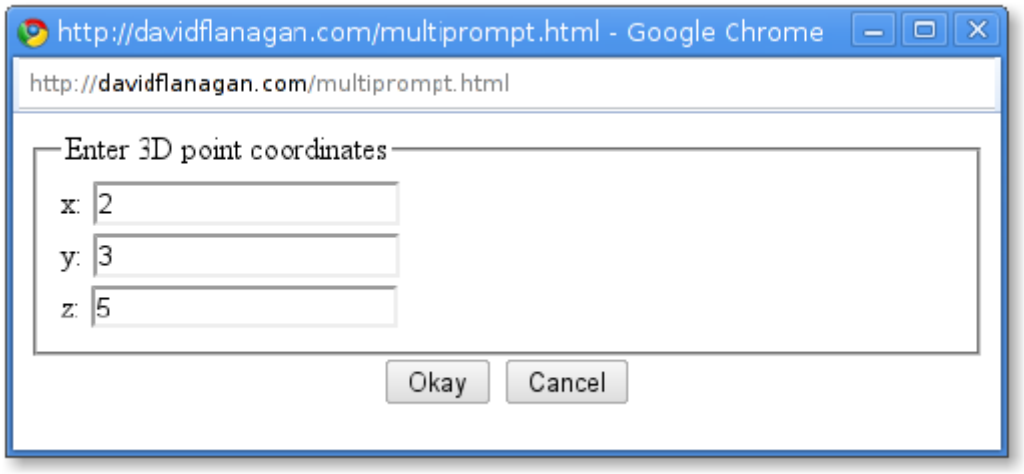
4.6. Mostrar en una nova finestra informació de l'exercici 4.4 (pensa que els navegadors solen bloquejar finestres emergents). En aquest exercici es toca:

- window.open().

EXTRA BONUS! Orgull de programador/a! Pujarà 1,5 punts d'aquesta pràctica el primer que descobreixi el error d'aquest codi font (ho explicaré a classe).

4.7. Ejemplo 14-4 del libro del rino, pág. 350: «An HTML file for use with showModalDialog()».

Se trata de construir un Modal Dialog -como el que se presenta en la figura- que contenga elementos HTML. En la página 349 del libro del rino explica el comportamiento de la función showModalDialog() del objeto Window. El código solución está explicaco en el libro.



Has de prestar atención a las indicaciones del libro.

- Your code is not a stand-alone HTML file.
- It must be invoked by showModalDialog().
- It expects window.dialogArguments to be an array of strings.
- The first element of the array is displayed at the top of the dialog.
- Each remaining element is a label for a single-line text input field.
- Returns an array of input field values when the user clicks Okay. Si el usuario presiona el botón «Cancel» el diálogo se cierra sin especificar un valor de retorno. En ambos casos has de cerrar la ventana: échale un ojo al epígrafe 14.8.1.1 «Closing windows» de libro.
- Use this file with code like this:

```
var p = showModalDialog("multiprompt.html",
                        ["Enter 3D point coordinates", "x", "y", "z"],
                        "dialogwidth:400; dialogheight:300; resizable:yes");
```

4.8. Si quieres tener clara la diferencia entre los distintos tipos de Modal Dialogs, escribe el código del ejemplo de la página 348-349 del libro del rino. Fíjate en qué métodos **bloquean**

la ejecución del resto del JS, qué devuelve cada uno de ellos y cuál de ellos es recomendable para **debugging**.

```
do {
        var name = prompt("What is your name?");                    // Get a string
        var correct = confirm("You entered '" + name + "'.\n" +     // Get a boolean
                              "Click Okay to proceed or Cancel to re-enter.");
} while(!correct)

alert("Hello, " + name);          // Display a plain message
```

4.9. El objeto **Location** representa la URL actual que se muestra en la ventana del navegador. Es muy útil **descomponer esta URL** en sus diferentes componentes -llamado URL , pues de ahí podemos extraer los argumentos en forma de par name-value de las variables que se enviarán al servidor para ser recogidas allí en un script.
La parte de la URL que contiene estos parámetros se recoge en la propiedad search.

El ejemplo 14-2 del libro del rino muestra la definición de una función de propósito general urlArgs() que puedes usar para extraer los argumentos de la propiedad search de una URL. El ejemplo usa decodeURIComponent(), que es una función global definida por client-side JavaScript (puedes echarle un ojo a Global en la referencia JS de la Part III del libro). Échale un ojo al código y luego intenta repetirlo, a tu estilo:

Example 14-2. Extracting arguments from the search string of a URL

```
/*
 * This function parses ampersand-separated name=value argument pairs from
 * the query string of the URL. It stores the name=value pairs in
 * properties of an object and returns that object. Use it like this:
 *
 * var args = urlArgs(); // Parse args from URL
 * var q = args.q || ""; // Use argument, if defined, or a default value
 * var n = args.n ? parseInt(args.n) : 10;
*/

function urlArgs() {
        var args = {};                                  // Start with an empty object
        var query = location.search.substring(1);       // Get query string, minus '?'
        var pairs = query.split("&");                   // Split at ampersands
        for(var i = 0; i < pairs.length; i++) {         // For each fragment
                var pos = pairs[i].indexOf('=');        // Look for "name=value"
                if (pos == -1) continue;                // If not found, skip it
                var name = pairs[i].substring(0,pos);   // Extract the name
                var value = pairs[i].substring(pos+1);  // Extract the value
                value = decodeURIComponent(value);      // Decode the value
                args[name] = value;                     // Store as a property
        }
        return args;                                    // Return the parsed arguments
}
```