Qué es una SPA

SPA es la sigla en inglés de: Aplicación de una Sola Pagina (SINGLE PAGE APPLICATION). Lo que define su nombre es que todo el contenido HTML de nuestra aplicación web, se resume en una única pagina o documento HTML.

Su principal ventaja es que, al cargar nuestro sitio web en el equipo del usuario, se descarga todo el HTML que conforma la web en sī. Al tener descargado el HTML, cuando necesitamos cambiar de sección dentro de la aplicación web, solo debemos invocar al bloque HTML correspondiente, y que el mismo se escriba en el tag HTML que oficia como contenedor.

VENTAJAS
Entre las ventajas que encontramos y destacamos de las aplicaciones SPA, podemos decir:

1. Rapidez en la renderización de contenido en pantalla
2. No tenemos que generar un documento HTML por cada sección de nuestra web
3. Evitamos tener que cargar todo el documento HTML cada vez que el usuario va hacia otra sección

DESVENTAJAS

1. A los buscadores se les complica indexar su contenido
2. Debemos hacer malabares para poder generar el historial de navegación y que esta responda a los botones ATRAS y ADELANTE del navegador web, si el usuario interactúa con estos
3. Rompe un poco con el paradigma de auditoria de LIGHTHOUSE para poder respetar el estándar de una aplicación web
4. En asociación con el punto anterior, CORE WEB VITALS también es otro punto que se ve afectado con la construcción de una SPA

Antiguamente se manejaba el método PUSHSTATE del objeto JS history, para agregar y eliminar contenido de navegación por las secciones de una SPA. Como esto causaba bastantes problemas porque las SPA no respondían del todo bien, JQUERY marco tendencia en la forma más moderna de desarrollarlas, implementando la interacción mediante el HASH generado con cada click de navegación.

Esto último se adaptó a JavaScript convencional, y es lo que hoy aprenderemos a dominar.

Estructura de una SPA

VARIABLES PRINCIPALES
Partiendo del ejemplo base que tenemos para desarrollar una SPA, migraremos todo el contenido HTML a plantillas desarrolladas en JS. Las mismas serán invocadas a partir de una función, la cual nos retornara simplemente el HTML de cada pagina.

Para poder contener toda la manipulación de datos que necesitamos hacer, definiremos un objeto JSON, el cual nos servirá para buscar el path que el usuario solicita desde el documento HTML, a través del menú de navegación, y con ese dato le diremos que acción debe tomar la aplicación web.

Por otro lado, en el único documento HTML tenemos un DIV definido con un ID=MAIN, donde escribiremos el HTML de la plantilla solicitada.

Desarrollemos esta estructura dentro del archivo denominado variables.js.

PLANTILLAS HTML
Una vez terminado esto, definimos las funciones que conformaran cada pagina HTML retornándonos el template en cuestion.

Para armar cada funcion() debemos CORTAR cada bloque HTML del documento principal, para luego pegarla en la función que corresponde.

Recordemos eliminar la CLASE “HIDE” del primer DIV de cada sección HTML.

LOGICA
Ahora debemos manejar la lógica de la aplicación. Por un lado, tenemos en el documento INDEX.HTML botones que ofician de menú de navegación. Estos botones tienen configurado el atributo HREF, y generan un HASH en la barra de navegación. Este HASH que generan hacen referencia a cada una de las secciones o paginas virtuales, que nuestra SPA contiene.

FUNCTION OBTENERHASHDEURL
El HASH generado en la barra de navegación es el que nosotros necesitamos para tener como referencia de qué plantilla debemos mostrar en la pantalla. Para poder acceder a dicho HASH, disponemos de una propiedad que toma su valor automáticamente desde la barra de URL, la cual esta contenida dentro del objeto JS Location. Sobre esta ruta contenida en la barra de URL, necesitamos tomar solo el nombre de la sección, eliminando el HASH que se genera junto a este.

Para realizar esto, debemos utilizar el método slice(), al

cual debemos indicarle como parámetro el valor 1, el cual limitara el corte a solo el primer carácter de la cadena. También, junto al método slice(), agregamos el método toLowerCase() el cual convertirá a minúsculas el resultado obtenido. Con esto armonizamos la posible escritura combinando mayusculas y minúsculas que pueda forzar el usuario y con esto ya controlamos ese posible error entre lo que ingresa el usuario y el mapeo de rutas que tenemos nosotros definido en el JSON.

A este resultado, lo retornamos a través del método que invoca esta linea.

FUNCTION ACCIONESSEGUNELPATH
Lo otro que debemos definir, es una función que reciba como parámetro la ruta obtenida del HASH de la barra URL, y nuestro objeto JSON de mapeo. De esta forma buscamos el path obtenido de la barra de url en el JSON, y retornamos a la función la acción que debemos realizar según el path detectado.

Si dentro del path tenemos una ruta no declarada, devolveremos ***undefined***, aprovechando la desestructuración (destructuring) de JAVASCRIPT.

FUNCTION MOSTRARVISTA (TITULO, COMPONENTE)
Ahora nos queda armar una función que será invocada cuando analizamos el ruteo que debe hacer nuestra aplicación. La misma recibirá dos parámetros (titulo y componente), y se ocupara de ajustar la propiedad document.title, y de mostrar en index.html la vista de la mano del template HTML que le indiquemos.

Por ultimo armamos una función denominada ruteo(), la cual aunara las funciones anteriores para:
1. Obtener el HASH de la barra de URL

2. Invocar al método de acciones según el path
3. Iniciar un proceso mediante la instrucción SWITCH, donde comparamos el CASE obtenido, para finalmente invocar una función **MOSTRARVISTA**, que muestre la VISTA esperada y que ajuste el titulo de la pestaña del navegador web.
4. En el caso que una ruta definida mediante el HASH, no exista en nuestro JSON de RUTAS, la sentencia SWITCH hara que caiga por defecto en la opción DEFAULT, la cual mostrara la vista dedicada a el famoso error 404.

APP FUNCIONAL: APP.JS

Nuestra SPA ya puede funcionar de forma optima. Solo nos queda disparar la función ruteo() que es la cual se ocupa de manejar todo el código que construimos. Para ello, debemos referenciar la función ruteo() en dos momentos importantes:

1. cuando se carga la pagina, mediante la escucha del evento **DOMContentLoaded**.
2. Cada vez que cambia el HASH de la barra de URL. Este evento lo podemos detectar mediante **hashchange**, el cual no esta activo en el objeto JS **document**, sino en el objeto JS **WINDOW**.

BONUS TRACK

Tenemos un XHR que nos carga y muestra una serie de productos contenidos en el archivo **products/products.json**. Dentro del ruteo de la sección productos, agregamos un **settimeout**() que nos dará tiempo a que el **TEMPLATE HTML** se cargue en pantalla, para luego invocar a la función **LISTARPRODUCTOSENVISTA**() y que muestre los productos que la empresa ficticia vende. Esta función espera dos parámetros:

1. El parámetro del elemento HTML donde se cargara la

lista de productos
2. El nombre del elemento HTML que tiene predefinido un LOADER, para poder borrar el mismo una vez cargados todos los productos en la pagina.