

TRABAJO PARA EL PRIMER TRIMESTRE DE LA ASIGNATURA DE DESARROLLO EN ENTORNO CLIENTE

ESTIMULADOR DE REFLEJOS MECÁNICOS

AUTOR: Moyano Fajardo, Daniel

DNI: 05964807X

ASIGNATURA: DESARROLLO EN ENTORNO CLIENTE

En Madrid, a día 18 de diciembre de 2023

# 

# Resumen

El proyecto “SUPER MENU” se trata de una oda a la interfaz de usuario del primer Super Hot y mostrar la cantidad de contenido que queda eclipsado por el juego que el usuario suele perderse al jugar el título, adicionalmente en la 2ª evaluación una vez terminado el subproyecto de treedude, un minijuego presente dentro del menú de Super Hot, se realizará un test de tipografía para ayudar a mejorar la mecanografía del usuario. Todos los apartados del menú y los propios juegos se desarrollaron usando el editor de código: Visual Studio Code con los lenguajes html, css y javascript. En un primer momento se desarrolló el proyecto para ser utilizado en ordenadores y portátiles.

Con todo esto, la meta final del proyecto es mostrar todo el contenido adicional del Super Hot original y que pasen un buen rato practicando mecanografía y jugando el minijuego de treedude. La idea final es imitar el comportamiento del treedude original con una barra de vida que vaya bajando más rápido según el jugador vaya obteniendo más puntos aumentando así la presión sobre el jugador, así mismo con el test de tipografía su meta será ayudar a mejorar la mecanografía del usuario.

Las puntuaciones de treedude serán guardadas en las cookies y si el usuario decide pagar podrá guardarlas en una bbdd mediante blockchain y una API externa, mientras que las puntuaciones del test de tipografía serán guardadas en el localStorage.

En resumen, el proyecto de SUPER MENU busca enseñar el contenido adicional presente en el juego de SUPER HOT, mejorar los reflejos del usuario mediante el minijuego de treedude y la mecanografía mediante el test de tipografía.

# Estado del arte

**Array**: Conjunto de elementos del mismo tipo que son agrupados.

**Clase**: En css una clase es la manera de referirse a un conjunto de elementos y aplicarles simultáneamente a todos el mismo estilo.

**Código ASCII**: O American Standard Code for Information Interchange, es un sistema codificado que asigna un valor numérico a diferentes caracteres utilizados en la comunicación electrónica. Fue desarrollado en la década de 1960 como un estándar para la transferencia de datos entre diferentes dispositivos informáticos.

**Divs**: Capas creadas por un programador para dividir la página en secciones.

**Etiqueta span**: Suele ser usada para delimitar zonas de textos y aplicarlos diferentes estilos o contextos.

**Función anónima**: Conjunto de acciones que no están enlazadas a un identificador.

**Función trim**: función que quita los espacios presentes al inicio de una cadena de caracteres (también conocida como string) y al final.

**Github**: plataforma donde uno puede crear repositorios de código y compartir proyectos de desarrollo con la comunidad al mismo tiempo que se posee un historial de versiones y modificaciones.

**Hacer un DOOM**: el DOOM original a pesar que parece estar hecho en 3D el mismo fue hecho completamente en 2D aumentando el tamaño de los sprites según el jugador se "acercaba") y replicar todo mediante código y manipulación de sprites.

**Interfaz de usuario**: medio que permite la comunicación entre el usuario y la máquina, equipo o dispositivo usado

**Mapa de bits:** Estructura o fichero de datos que representa una imagen de forma digital, en el que las dimensiones están determinadas por la cantidad de píxeles horizontales y verticales que componen a cada imagen

**Mecanografía:** Habilidad para introducir escribir mediante un teclado, los tests de mecanografía sirven para hacer que aumente la precisión y la velocidad en la que el sujeto pueda ser capaz de introducir información mediante el teclado.

**Paleta de colores:** Conjunto de colores usados en este caso en la estética de una página web

**setInterval:** Función que se activa en intervalos marcados por una cantidad de tiempo indicada por el programador y es usada para reproducir código cada X tiempo.

**Shooter:** Refiriéndonos al ámbito de los videojuegos un shooter es aquel cuya misión principal es empuñar un arma de fuego como por ejemplo una pistola y normalmente acabar con todos los enemigos en pantalla

**Sprites:** Pequeño mapa de bits que es dibujado en la pantalla a partir de un dibujo inicial, en el ámbito de los videojuegos se define Sprite como un conjunto de imágenes que representa un personaje, objeto o una parte de ellos de manera gráfica y que se utiliza para poder crear cualquier efecto de movimiento o para cambiar su estado o posición en la escena

**SUPER HOT:** El segundo Shooter independiente más innovador que he jugado en años, siendo el 1º ULTRAKILL (y DOOM ETERNAL dentro de los triples A), la mecánica principal es que el mundo se mueve a una velocidad directamente proporcional al movimiento del jugador, si el jugador se queda quieto el juego irá a una cámara lenta tan alta que parecerá que no se mueve y si se mueve rápido el juego se verá a velocidad normal

**Triple A:** Se refiere a juegos desarrollados por grandes empresas de videojuegos como pueden ser Nintendo o SEGA. Siendo el caso opuesto a los juegos independientes, los cuales son creados por equipos más reducidos de personas, ejemplos de este último tipo son juegos como Super Hot y/o Undertale

Tabla de contenido

[**INTRODUCCIÓN 4**](#_heading=h.gjdgxs)

[**DESARROLLO 6**](#_heading=h.30j0zll)

[PANTALLA DE INICIO 6](#_heading=h.1fob9te)

[LA ANIMACIÓN DE LAS LETRAS DEL INICIO 6](#_heading=h.3znysh7)

[LA INTERACCIÓN DEL USUARIO 8](#_heading=h.2et92p0)

[Comenzando con el menú 9](#_heading=h.tyjcwt)

[Haciendo un reloj 11](#_heading=h.1t3h5sf)

[Nuestro puntero para seleccionar opciones 11](#_heading=h.4d34og8)

[Creando el menú de GAMES 14](#_heading=h.2s8eyo1)

[Desarrollo de por fin, TREEDUDE.”exe” 15](#_heading=h.17dp8vu)

[Los SPRITES DEL LEÑADOR 15](#_heading=h.3rdcrjn)

[Diseñando el escenario 16](#_heading=h.26in1rg)

[La generación aleatoria 17](#_heading=h.lnxbz9)

[UNNUS ANNUS MEMENTO MORI 18](#_heading=h.35nkun2)

[**CONCLUSIONES 19**](#_heading=h.1ksv4uv)

[LIBRO DE BUGS 20](#_heading=h.44sinio)

[**BIBLIOGRAFÍA 21**](#_heading=h.2jxsxqh)

[**Apéndices 21**](#_heading=h.z337ya)

# INTRODUCCIÓN

La idea inicial era la de crear un selector de juegos entre dos proyectos, el Estimulador de reflejos mecánicos también conocido como el minijuego treedude dentro del grandísimo juego SUPER HOT. Y el entregable de la primera evaluación y el test de mecanografía conocido como typingtest para la 2ª. Dicha idea evolucionó a replicar el menú de SUPER HOT para dar importancia a la increíble Interfaz del usuario del juego que a menudo queda eclipsada por la genialidad del propio título.

Debido a esto el uso del ratón quedará limitado en un principio solo al typingtest, e incluso se verá si también es eliminado de dicha zona, por otro lado, sí estará disponible un ratón propio en ciertos simuladores de otros apartados del menú. En el treedude la dificultad principal estará ubicada en la animación del juego que deberá ser hecha con puro código ASCII.

Luego la parte de la mecánica del juego será algo más sencilla: por un lado, se tendrá la vida que irá bajando poco a poco a no ser que el leñador treedude tale el árbol, cada vez que el árbol sea talado se bajará una posición. El árbol tendrá dos lados en los que se puede generar una rama, si en algún momento rama y leñador colisionan será Game Over para el leñador, parte del desafío será generar las ramas aleatoriamente en un lado u el otro. Por otro lado, se encuentra la parte del Typing test: consistirá en lo siguiente.

Mediante ya se verá si un array o un fichero json externo habrá una larga lista de palabras. Se generarán unas primeras 6\*6 números aleatorios del 0 - al nº de palabras del fichero para seleccionar las palabras de manera aleatoria. La idea es calcular las palabras \* minuto y pulsaciones \* por minuto del usuario en un tiempo de 1 minuto. cuando el usuario complete una línea de palabras se generarán abajo del todo una ristra nueva de seis palabras, la cuenta atrás iniciará con la primera pulsación del usuario sea correcta o incorrecta.

Para finalizar como oda a una buena interfaz de usuario se intentará replicar el menú del superhot, en un principio el único directorio que no tendrá funcionalidades será el de settings, no obstante, será incluido y representado también el resto se tratará de replicar cuanto más parecido posible mediante código evitando el uso de videos siempre que sea posible, como por ejemplo VR donde se intentará hacer un DOOM.

Debido a las posibles dificultades que este aumento puede suponer se irá avanzando en la parte del menú paulatinamente según se vaya avanzando en los proyectos originales, véase treedude y typingTest. En un primer momento durante las versiones primeras se usará la fuente por defecto del navegador a la de programar, será una vez alcanzado una serie de mínimos entregables antes de la fecha de entrega que se barajará la posibilidad de cambiar la fuente por una más similar a la vista en el juego de SUPER HOT.

A pesar de intentar replicar la interfaz de usuario presente en el superhot, se optará por paletas de colores semejantes, más no las mismas que las vistas en ese juego, como ejemplo tenemos el fondo que en vez de ser completamente negro es un azul marino muy apagado.

Todas las desventuras sufridas a lo largo de la realización del proyecto serán anotadas en este mismo documento y les será dado un lavado de cara previo a la entrega de la documentación, márgenes, estilos, limitación de apartados, cambios a un léxico más profesional, evitar repeticiones, introducir un orden estricto cronológico

El proyecto va dirigido a personas y jugadores de todas las edades, aunque seguramente sean los de edades más avanzadas los que apreciarán de mejor manera el aspecto escogido para la implementación de la interfaz del proyecto, otro detalle que tendrá el público destino del proyecto será una escasa sociabilidad y ganas de autosuperación

# DESARROLLO

## PANTALLA DE INICIO

Como cualquier juego se necesitaba una página donde comenzase la acción, se crea el árbol de directorios básicos un index.html, style.css y action.js. Se comienza introduciendo en medio de la pantalla un mensaje con el siguiente texto:

PRESIONE CUALQUIER TECLA

Con el texto ya colocado se añade el color de fondo: un apagado tono de azul cercano al negro. En primer momento se decide iluminar el texto con un sombreado amarillo, pero se decide que queda algo corto, por lo que se decide alumbrar de una manera alternativa y poco convencional. Se duplica el mensaje con las mismas propiedades, pero se le añade una clase adicional. Lightning

Con la propiedad de position absolute que hace que no ocupe ningún lugar en el esquema de la pantalla y por lo tanto no empuje al su original y se le da una fuente amarilla en contraposición a la fuente blanca de su original, luego se difuminó el duplicado y se le dio algo más de sustancia mediante hacer la fuente en negrita.

### LA ANIMACIÓN DE LAS LETRAS DEL INICIO

Para aportar un efecto sorprende y fresco a la pantalla de inicio se decidió implementar animaciones a las distintas letras del mensaje de inicio, primero fue necesario obtener la longitud del mensaje, una vez que se disponía de dicho dato se creó un array donde serán guardados los distintos intervalos de desaparición de cada letra, para esta funcionalidad se trató hacer uso de una función anónima, pero no pudo usarse.

Llegados a este punto se cambió el enfoque para realizarlo con un bucle que sería reproducido el número de veces especificado por una variable i y con cada iteración del bucle se colocará en la posición marcada por otra variable “j” que irá aumentando hasta alcanzar el valor de la variable i: un número aleatorio entre 1000 y 500.Este será el intervalo con el cual se activará el parpadeo. Al principio ingenuamente se trató de acceder a cada letra del mensaje y cambiarla el color de manera directa… Cosa que era imposible.

#### LA ANIMACIÓN DE LAS LETRAS DEL INICIO: UN NUEVO ENFOQUE

Para realizar las animaciones se tuvo que dar una clase individual a cada letra del 0 a la longitud del mensaje y en dicha clase sí. Alterar sus propiedades. Para probar el setInterval, primero se probó el llamar a una función con parámetros a modo de prueba ya que no sería la primera vez que funciones similares tienen problemas al llamar a otras funciones y estas poseen parámetros. Como en otros casos esto resultó en error, tras horas de investigación se dio con la solución, para poder realizar esta tarea sería necesario una función anónima.

##### \* You called for help… And the Anonymous function came

Desde el setInterval llamaría a la función anónima la cual no contaría con parámetros. Significando que el setInterval funcionaría a la perfección y la función anónima sería la cual llamaría a la función que necesitaba los parámetros especificados. Fue así como se solucionó el problema

Disponiendo ya de los parámetros en el setInterval ahora se necesitaba poder afectar a varios elementos al mismo tiempo con la función. De una no se podía por lo que se tuvo que hacer con un bucle que recorriera los distintos elementos con la misma clase

De primeras se probó con un bucle for in de los elementos de la clase especificada, el problema era que ejecutaba el código una vez más de la necesaria, cambiando el enfoque a algo más simple se saca la longitud del resultado de sacar todos los elementos de la clase especificada y se recorre desde el 0 dejando de ejecutar el código al llegar a la longitud especificada.

Una vez se resolvió el tema de afectar a dos o más elementos a la vez tocaba cambiar la propiedad que era necesaria, comenzando por el color de la fuente. Para ello se tuvo que descubrir el formato en el que se guardaba el color, siendo este rgb(x,y,z); Ahora tocaba adaptar el código con lo nuevo cambiado y aplicado teniendo que llamar a las clases individuales una a una.

Primero se tuvo que sacar el tamaño del mensaje sin espacios ya que los mismos no eran contados en la animación. Para sacarlo se añadió al mensaje original, el identificador “modelo”, la cosa se complicó al usar etiquetas span para cada una de las letras la longitud dada por la cadena era mayor a lo que realmente sería, por lo que se tuvo que clonar por tercera vez el mensaje poniendo el 3º en la misma posición, pero oculto y position absolute, el mismo solo tendría la cadena tal cual, sin span ni nada, con un css similar al de la clase lightning pero sumándole visibility: hidden para que no se muestre nunca. Luego se tuvieron que quitar los espacios internos por lo que trim no nos valía.

Se encontró la función replace con el siguiente parámetro /\s/g, ''. Con esto ya obtuvimos el tamaño de la cadena sin espacios. El siguiente paso era el de llamar a la función que nos daba los tiempos del intervalo por cada letra. Tocaba un bucle.

Se cambió en el css el formato de los colores para encajar con el de JavaScript. Ahora comenzamos con una condición para diferenciar si se trata del h1 original o del h1 sombra.

Al estar ahora modificando cada clase y su duplicado en concreto, el modificar el color de fuentes a distinto en función, de si su padre es original u sombra alternará entre dos colores distintos, blanco para el original cuando estaba encendido o amarillo para la sombra cuando estaba encendido y negro ambos cuando estaba apagado.

Sin embargo, en distintas pruebas se vio que quedaba mejor estéticamente si ambos se les daba el color blanco cuando estaban encendidos por lo que finalmente se fue por dicho camino.

### LA INTERACCIÓN DEL USUARIO

Una vez se dio por concluida la parte visual de la pantalla de inicio se procedió a la interacción.  Se comenzó creando un directorio con los ficheros index.html, style.css y action.js en el directorio menú. La tarea de ese momento consistirá en mediante js al pulsar cualquier tecla se envíe al usuario a la página siguiente. Primero se agregó el event listener al body

A continuación, tocaría averiguar cómo acceder a la nueva página usando javascript, con la siguiente línea se cambia la ruta de ejecución: Y con esto se dio por finalizada la versión 1 de la pantalla de inicio.

## Comenzando con el menú

Comenzando con el menú se establece el mismo color de fondo que la pantalla de inicio y la página fue dividida en 3 divs: uno padre y 2 hijos dependientes del 1º. La idea surgida fue la de no poder navegar con el ratón en esta pantalla y obligar al usuario a usar el teclado para navegar entre las distintas opciones.

De manera temporal al div padre se le da un width del 70% y una altura de 97vh. El menú que trataremos de replicar en la V1 será el inglés original, comenzando con la siguiente pantalla



*Ilustración 1 Vista del modelo a replicar*

La parte superior: LEVELS, ENDLESS, CHALLENGES, README, REPLAYS, RECRUIT, CREDITS y more en un principio serán eliminadas, mientras que la parte de quit.exe, será sustituida por /.., la ubicación marcada abajo será C:\MENU\, siendo la pantalla anterior la de inicio la supuesta C:\ . Como primer intento se dispone el método de montaje de la página a grid, dividiéndolo en 3 columnas iguales.

Al primer div se le da un grid -column: 1 / span 2, al segundo no se le especifica nada quedándose en la posición 3, con un span de 1. Y por último al 3º le damos un grid-column: 4 / span 4. Para diferenciarlos se les da fondo: azul al primer div (el de más a la izquierda), transparente al 2º y rojo al 3º.

Para poner la parte de de abajo se decide reducir la altura del div padre a 90vh y generar otro div de la misma anchura, pero de 7vh sin margénes justo debajo que será el que lleve la parte de la ruta. Para quitar y aplicar los nulos márgenes arriba y abajo, pero mantenerlos a la izda y la derecha se modifica el margin a: 0 auto; aplicando 0 márgenes arriba y abajo, pero los automáticos a la derecha y la izquierda. Para poder visualizar temporalmente el nuevo div le se le da el fondo white.

Se le aplica el borde blanco igual que a los demás y se continúa. Se quitan los colores de fondo y se les da a los hijos del container mediante #container > div. El borde blanco presente en los padres para delimitarlos. Primero se rellenarán los div estáticos de este menú siendo los dos de la izquierda y el de abajo del todo.

El primero que ponemos el /.., por el momento se creará en una etiqueta p con clase: "goBack". En el div de en medio como en el juego introducimos lo siguiente en otra etiqueta p: <UP-FOL>, se le dará la misma clase que el p del primer div, pero para que no confunda los símbolos de menorque y mayorque con una etiqueta se tendrán que introducir mediante su código: %lt y &gt, como queda muy pegado se le aplica a la etiqueta p en general temporalmente un margin-left de 10px;

A continuación, solo quedarían los directorios: SETTINGS, APPS, DEMOS, CELLULAR, WIRES, GAMES, VR, ART y VIDEOS. Junto a sus respectivos >FOLDER< en el 2º div, cada una con su clase go<Directorio> Luego se introdujo el C:\ en el div inferior. Lo siguiente que se tuvo que hacer fue introducir los elementos posicionados de forma absoluta en las esquinas situadas a la derecha la de arriba con la hora y la de abajo con el "sistema operativo" comenzamos con la sencilla, la de abajo.

Se creó una clase general para posicionar elementos de manera absoluta "abs", y para posicionar el ID SO y el background color el mismo que el del body para ocultar la línea del borde que ocupa y que quede igual que en el menú original. Para la hora se creó un segundo p con clase abs e id time que se colocó en la parte de arriba y en la misma posición de izquierda/ derecha que el p de SO. Una vez se terminó de colocar los elementos estéticamente llegaba la hora de darle funcionalidad comenzando por la hora.

### Haciendo un reloj

Primero será necesaria una constante con la fecha y horas del día en el que se esté ejecutando el programa. Esto se obtiene con const <nombre variable> = new Date(), que por defecto sin parámetros añadidos coge la fecha del sistema.

Para aumentar la limpieza del código la función del reloj será alojada en un subdirectorio aparte con el script time, se obtiene la hora y los minutos mediante la propiedad getHours() y getMinutes(), luego para darle un formato más arreglado se quiere que cuando los minutos sean de un solo dígito, le añada un 0 adicional a la parte delantera del número, esto se hará mediante la función addCero(n) la cual recibirá los minutos y en caso de que sea < 10, le añadirá como una cadena un 0 delante. Luego la cadena restante será devuelta para ser utilizada donde se la llame

Desde el script de actions se crea la función de update que será activada en un intervalo de 1000 milisegundos la cual igualará el valor del elemento con ID time el valor devuelto por nuestra función timePasses, es decir, la hora concatenada con los minutos.

### Nuestro puntero para seleccionar opciones

El siguiente apartado que fue atacado fue el de la selección, y para ello se creó otra clase en el css llamada selected, la cual mediante jQuery añadiremos o quitaremos según este seleccionada esa opción o no.

Por defecto al iniciar la página la opción seleccionada será la que esté situada la 1ª pero en versiones posteriores la opción seleccionada será aquella en la de la que anteriormente volviste, es decir, si se accede al directorio de APPS, pero luego vuelves, te debería seleccionar por defecto a APPS.

Mediante addClass(.<clase>) se agrega la clase de selected a los elementos, pero al ver el resultado se decide cambiar los elementos absolutos por la etiqueta span, lo cual obliga a modificar sus posicionamientos. Una vez recolocados se ajusta el margin-bottom de los p para que la selección se vea igual que en el producto original, véase pegados

De manera temporal en actions.js se comienza a crear dos arrays, uno con cada nombre de las distintas clases seleccionables y otro con si está seleccionado o no. Ambos serán metidos dentro de un 3er array "globalSelector" y mediante un bucle de sus elementos donde se juntarán, el de clases y el de selección en el array bidimensional globalSelector en la posición de i.

Una vez ya fue creado el array global de modelo se prepararon las funciones de avanzar una opción hacia arriba o hacia abajo que serán usadas cuando se pulse una tecla arriba o abajo. Comenzando con la función de avanzar hacia arriba lo primero será saber cuál es la opción que está activada véase la casilla  
globalSelector[i][1] == true. Una vez se obtiene la opción activada deberemos desactivarla, esto lo se hizo mediante el removeClass

Si la opción activada era la 1ª se situará en la última opción, de lo contrario se restará a la posición activada 1. Ya se sabe la posición nueva que deberá ser activada pero aún tenemos el modelo desactualizado, eso habrá que solucionarlo con un bucle.

Recorremos el array global desde 0 a la longitud del mini array focus, si i==j entonces se igualará a true el modelo, de lo contrario se igualará a falso la posición [j][1]. Por último, usando el modelo se vuelve a buscar el valor activado y mediante jquery se le añade la clase selected a los elementos con la clase de la posición [i][0], nos llevamos las funciones al directorio de additionalScripts y creamos el fichero Iselectores, desde el action llamaremos a optionUp y se le deberá pasar el array de las clases del fichero.

Ya finalizadas las funciones, llamamos a la función cuando se pulse la tecla de arriba mediante event listener al body. Al solo pasar las clases solo se efectuará la función una vez por lo que pasamos las clases y sus valores de verdadero o falso. No se consigue el resultado deseado por lo que deberemos hacer el array global en actions.js y mandarlo como parámetro junto clases. Una vez hechos los cambios debidos se consigue la acción deseada

Ahora tocaría con el código usado para optionUp hacer el código de optionDown, a tener en cuenta sobre los cambios necesarios estaría el en vez de reducir el valor de la posición del seleccionado se tendrá que incrementar, y cuando el valor sea 1 menos que la longitud del array (véase el último del array), se igualará a -1 para luego aumentarlo en 1 como a todos los demás y que vaya al primer elemento del array usado de modelo para las opciones.

Una vez terminado el código añadimos en el if del eventlistener del teclado un else if si la tecla pulsada es la flecha hacia abajo. Lo siguiente de lo que se ocupó el equipo fue el actualizar el tercer div, desgraciadamente en el momento en el que se pusieron a ello no tenía el material original a mano, por lo que de manera temporal se dispusieron a actualizarlo de una manera básica: para los diferentes directorios:

directory:<nombre de directorio> (tabulación) |>FOLDER<

Y para "/.." directory: GO UP

Para esto se tuvo de aumentar el array bidimensional a Tetradimensional, nombre de la clase, estado, tipo de "archivo": si es /.., fichero o archivo, para esta primera pantalla, si es /.. o fichero, en caso de ser fichero, estaría el último campo, nombre del fichero o directorio.

En los eventkey añadiremos un igualador del interior del html, debajo del valor de globalSelector actualizado. Con el valor devuelto por la función Update3(tetracadena,clases.length). Aunque previo en la 1ª ejecución desde action se igualará el valor interno de html del 3er div por el valor de /.. que es por el momento el valor por defecto al iniciar la página. Esto se realizó mediante jquery

<variable> = $(#summary);  
<variable>.html("<span>directory: GO UP</span>");

Se le aplica un padding al div summary (el 3º) para que no quede tan junto y se ajusta #SO para que quede algo mejor colocado. Ya que el actualizar el div 3º también tiene que ver con la selección también se incluye la función de Update3 en Iselectores.

Primero Update3 tiene que saber cuál es elemento seleccionado, por lo que se invoca a la función previamente hecha que coincidentemente cuenta con los mismos parámetros, el valor que devuelve lo guardamos en una variable. Si el campo elegido tiene el valor /.. en el tipo devolverá:

“<span>directory: GO UP</span>”

Y cómo de momento no se tienen ficheros y solo directorios solo se tomará como otra posibilidad que sea un directorio de forma temporal else devolverá:

"<span>directory:" + cadenaTetradimensional[elegido][3] + "&emsp;|FOLDER</span>";

Al no estar del todo de acuerdo como se termina representando se crean dos clases adicionales flleft y flright con las propiedades de float left y right respectivamente a continuación se modifica el código devuelto en caso de no ser /..

"<span class='flleft'>directory:" + cadenaTetramensional[n][3] + "</span><span class='flright'>|FOLDER</span>"

Una vez ya se tiene el código aplicado en la tecla de pulsación arriba lo llamamos también al pulsar abajo. Y con esto se comienza la codificación de la pulsación de la tecla enter. Se creará una función más en Iselectores navegation(cadenaTetradimensional, longitud de clases)

Comenzando como la anterior función lo primero es averiguar cuál es la opición seleccionada, para eso se usa findTrue, junto los mismos parámetros que recibió la función. Si el campo elegido tiene el valor /.. en el tipo devolverá:

".."

de lo contrario devolverá cadenaTetramensional[n][3]. Esto será completado en action que estará esperando el resultado de navegation para concatenar al final del valor

"/index.html"

Ahora antes de proseguir con las pruebas la situación obliga a crear la primera tanda de árbol de directorios todos ellos respetando la estructura index.html, style.css y action.js. Se procede a mejorar la interfaz del menú mediante el uso del flex. Se continúa yendo directamente al directorio importante GAMES.

## Creando el menú de GAMES

Se empieza fusilando la base del menú para adaptarlo al menú de GAMES, aquí se encuentran los elementos FILES, por lo que se actualiza la función de Iselectores Update3, haciendo ahora la distinción entre es /.., directorio u otra cosa, en este caso ficheros/archivos. En el último caso temporalmente solo devolverá:

 "<span class='flleft'>app:" + cadenaTetramensional[n][3] + "</span><span class='flright'>|--FILE->#</span>".

En el producto original el "#" era un código numérico, pero al en este punto no tener identificadores para los archivos lo dejaremos como una incógnita y a tener en cuenta para futuras versiones.

En el modelo original se tenía a treedude.exe y shrl.exe, en nuestra versión shrl.exe será sustituido por nuestro "typingtest.exe", se redirigen las rutas de los scripts y para finalizar, se actualiza la capa de route a la ubicación de MENU\GAMES. Y al igual que se hizo con el html fusilamos también el css y js para ser usado de base para el código js, el css es copiado tal cual

Comenzando ahora sí con los cambios se modifica el array de clases, ahora solo teniendo 3 elementos: goBack, goTreeDude y goTypingTest, debido a que focus es construido dependiendo de clases, el mismo ya se ve modificado automáticamente.

Otros que sufren modificaciones son los arrays de tipo y nombre tipo ahora solo teniendo 3 elementos /.., archivo y archivo y nombre solo teniendo: "","treeDude" y "typingTest" todo lo demás se queda igual. Se prepara el árbol de directorios de games y se comienzan las pruebas

## Desarrollo de por fin, TREEDUDE.”exe”

### Los sprites del leñador

Se descubre en GitHub que alguien hizo el minijuego de treeDude en C, es usado como referencia de momento para realizar los sprites. Se crea en aditionalScripts draw.js que será quien tenga las funciones de dibujar los sprites.

Para hacer las funciones que pasarán las cadenas con los dibujos se hace uso de las funciones con flecha sin parámetros al ser html los espacios serán sustituidos &nbsp; y debajo del código mediante comentarios se muestra el resultado final

Primero se realizan los sprites del talador quieto tanto posicionado a la izquierda como la derecha mediante un array constante que se devuelve en una función seguimos haciendo el sprite del talador cortando por la izquierda, este se realiza usando de base el talador quieto situado a la derecha ya que es el que más se asemeja

Una vez se terminó el sprite tocaba alinearlo y colocarlo un poco más arriba que el quieto y hacer la sensación del saltito del juego original. Para la sensación de salto simplemente se agregó una fila adicional vacía y así dar la impresión de que está más alto.

Primero para alinearlo se prueba poniendo más espacios en lado izquierdo, no funciona y de poner más espacios en la derecha se aleja más, por lo que se decide hacer más anchos los sprites del talador quieto, tampoco funciona, se vuelve a la idea original de en el cutingleft añadir más espacios, solo que colocar el muñeco en el flex-strat, esta acción da resultado. Se prosigue con el fellercutingRight se coge como base el iddleLeft y se añade la celda extra para la sensación de salto.

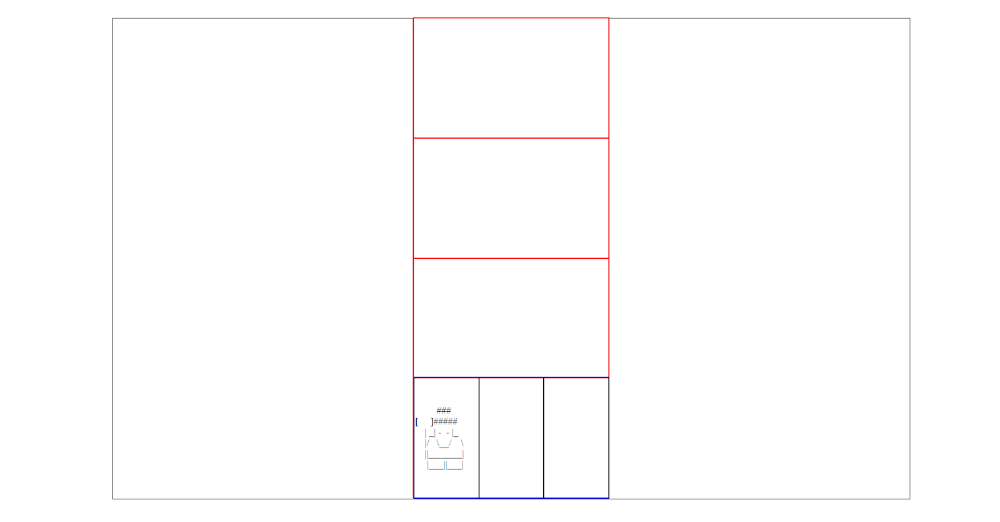
Tras añadir el suficiente nº de espacios en el iddle right se consigue que apesar de estar colocadas en el justify-content: center, que ambos sprites estén alineados. Se reintenta con el de la izquierda hasta conseguir el resultado deseado.

### Diseñando el escenario

Antes de continuar con los sprites se procede a diseñar la ventana donde se mostrará todo. Para hacerlo se coge el código de container y “supahContainer” de páginas anteriores. Luego para visualizarlo le damos un borde visible a container. Una vez ya tenemos el marco comenzamos a colocar los elementos de posición

Primero se coloca el “containerDude” quien lleva la posición del leñador, lo colocamos abajo del todo en medio, con position absolute para que no ocupe un espacio y ese mismo pueda ser ocupado simultáneamente por otro. Ahora toca colocar el “containerTree”, una vez tenemos una colocación temporal se continúa con los sprites

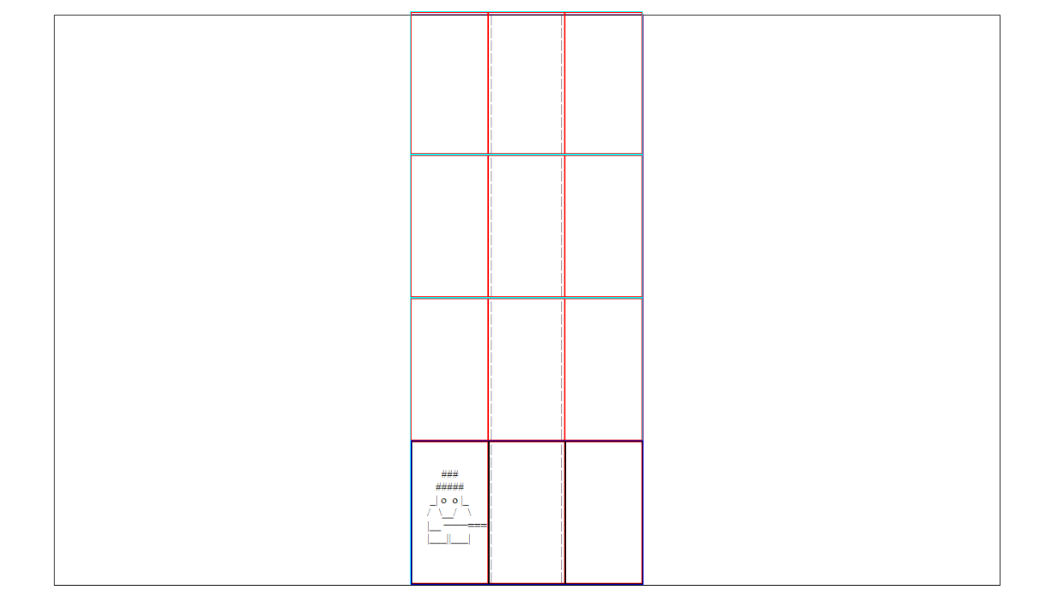
Vista:



*Ilustración 2 Vista de containerDude y containerTree*

En un principio el tronco se realizará sin adornos, siendo más similar a una tubería, en futuras versiones es muy probable que se mejore. Una vez se coloca el primer tronco y queda ajustado, dividimos las celdas superiores también y se coloca un tronco en todas las celdas del medio

Vista:



*Ilustración 3 Vista de containerTree con el propio árbol*

Se prosigue con la creación de los sprites de las ramas y se prosigue con la animación del movimiento del leñador para que pase del de cutting a iddle cuando suelte la tecla.

### La generación aleatoria

Ya está terminada la animación del leñador hecha ahora toca generar la "copa" del árbol aleatoriamente, véase la celda más superior. Primero generamos dos nº aleatorios entre 0 y 1. Si el primero sale 1 no podemos permitir que el segundo también salga 1 por lo que manualmente lo ponemos a cero.

Ahora se crea otro array este unidimensional de dos elementos para saber dónde está nuestro leñador. Si la última tecla que se pulsó fue la izda el primer campo será 1 y el 2º será 0, si fue la derecha el 1º será 0 y el 2º 1. Como por defecto el leñador siempre comienza en la izda. Creamos el array directamente con 1 y 0.

Ahora toca representar lo indicado por el modelo en el árbol primero debemos de recorrer el array bidimensional y tener en cuenta de si estamos en la celda de la izquierda o la derecha, así como si tiene o una rama en el modelo. Mediante la concatenación de los valores de J e I + las etiquetas del padre e hijo se puede seleccionar la celda correspondiente.

El siguiente paso que realizamos es llevar la función de pintarArbol a el fichero de draw.js. La de generarnuevo0 al no tener relación con la ilustración de modelos, la dejamos dentro de action.js, aunque cuando vayamos teniendo más funciones de mecánicas se terminará creando un fichero extra donde guardar dichas funciones aparte. De momento toca agregar la función de talar el modelo para luego replicarlo en la ilustración, para nuestra función talar le pasaremos las dos cadenas la del árbol y la de la posición del talador.

Primero, se recorre el array del árbol desde la longitud de árbol -1 al 0 pasando el valor a de [i] a [i-1] tras ello nos disponemos a hacer uso de generarNuevo0 para sustituir el primer valor, pero nos da error, teniendo que revisar cómo solucionarlo.

Se configura para que no se le pase ningún parámetro pero que devuelva un array de dos elementos el cual será guardado en la posición del array del árbol[0] que sería lo mismo que la posición [0] del array devuelto se guarde en el árbol[0][0] y la posición [1] en el [1][1].

Tras aplicar estos cambios al primer modelo igualamos el valor de su posición [0] a generarNuevo0. Después se llama a la función de pintar árbol para que teniendo en cuenta la cadena bidimensional pinte el árbol actualizado.

### UNNUS ANNUS MEMENTO MORI

Lo siguiente que se realizará será comprobar cuando el leñador y una rama "colisionan". Como se sabe es el último bloque de arrays el que contiene las ramas que están a la altura del leñador.

Por lo que con un "if" se revisa si el array del modelo del árbol[longitud array][n] y si la posición del leñador[n] en ambos se encuentra un 1 simultáneamente de ser así de momento, que nos avise por la consola, donde N será la posición del array a comprobar

Una vez conseguimos detectar correctamente las colisiones toca crear desde cero al no tenerlo en proyecto de GitHub el sprite de la tumba. Una vez hecha ya la tumba, tenemos que colocarla o en la posición en la que estaba el leñador o la rama, para facilitarnos las cosas se colocará la tumba en la posición del leñador y se borrará la rama.

Primero usando de base el código usado para dibujar al leñador en la izquierda o la derecha es usado para pintar la tumba en función de si el modelo de la posición del leñador estaba situado a la izquierda o la derecha.

Y luego se sobrescribe el valor 1 de la posición[i] a 0,0 y volvemos a invocar la función de pintar el árbol para que borre la rama sobrante. Lo siguiente es quitar todos los eventListener existentes por el momento siendo estos pulsar tecla izquierda, pulsar tecla derecha y soltar las respectivas teclas.

Al hacerlo con JavaScript mediante el removeEventlisterner no se consigue el resultado deseado por lo que se prueba mediante jQuery el método.unbind en una función. Al crear el eventListener con JavaScript y quitarlo con jQuery no parece funcionar. Se intenta realizar el eventListener también desde jQuery

Una vez lo cambiamos funciona a la perfección, lo siguiente que es realizado es dar unas proporciones fijas y mínimas a la ventana del juego y las propiedades del div para que al cambiar el tamaño de la ventana no se destroce la estructura del juego

# CONCLUSIONES

Teniendo en cuenta que no se pudo ni si quiera llegar a los mínimos establecidos por el propio grupo:

1. Creación estética de la pantalla de inicio (hecho)
2. Generación de números aleatorios (hecho)
3. Implementación de la animación de las letras que desaparecen en intervalos aleatorios (hecho)
4. Creación estética del menú principal (hecho)
5. Crear interacción menú principal con teclado para quitar el ratón (hecho)
6. Creación de los gráficos v1 del leñador (mediante ascii) (hecho)
7. Implementar los gráficos dentro de la ventana (hecho)
8. Implementar movimiento del leñador (hecho)
9. Implementar animación de corte (hecho)
10. Generación de las ramas de manera aleatoria (hecho)
11. Implementar muerte (hecho)
12. Mostrar puntos y crear puntos (en proceso)
13. Poner textos de la ventana
14. salir, (salir se puede en cualquier momento)
15. Aplicar estética al juego
16. forzar tamaño fijo para que no se desborde o se contraiga afectando a la vista del juego
17. Opción de volver a jugar
18. guardar puntos y mostrar ranking al final del juego, si la puntuación está entre los 10 primeros se guarda
19. Implementar pantalla de muerte

---------- MÍNIMOS DEL GRUPO

1. creación visual de la intro
2. Implementar cuenta atrás de la vida y la recuperación de la misma
3. Implementar aumento de velocidad, así como el mayor gasto de vida
4. Aplicar efectos de sonido (al navegar por las opciones)
5. Implementar música y efectos de sonido (juego)

Aún le falta al equipo de Neutral Face.inc algo más de tiempo para poder dar un producto aún mejor, no obstante, teniendo en cuenta las limitaciones técnicas y temporales consideran que han podido mostrar un desarrollo más que decente por lo menos en las partes de HTML, css y JavaScript siendo esta rama la más fuerte del “árbol”

Respecto al rendimiento del equipo ninguno tiene ninguna queja todos aportaron todo lo que pudieron y trabajaron de manera equitativa, para la segunda evaluación es casi un hecho que el proyecto de ESTIMULADOR DE REFLEJOS MECÁNICOS estará finalizado y seguramente con suerte alguno de los bugs apuntados en el libro de bugs será solucionado:

## LIBRO DE BUGS

* Al redimensionar el menú y hacerlo más pequeño la barra de localización se mueve al tener como referencia, al contrario que las p el tamaño de la ventana.
* Una posible forma de solucionarlo sería haciendo de su posicionamiento absoluto, pegarlo abajo del todo en la misma ubicación... arreglado más o menos
* Al cambiar el texto en el div summary el tamaño del div cambia, se debería de poner un width fijo y que no se expandiera más, pero de momento se hace la vista gorda.

La duda está en si dará tiempo en acabar el test de mecanografía y el menú al completo, solo el tiempo lo dirá.

# BIBLIOGRAFÍA

Para dudas generales se utilizaron:

*W3Schools online web tutorials*. (s. f.). <https://www.w3schools.com/>

GeeksforGeeks. (s. f.). *GeeksforGeeks | a computer science portal for geeks*. <https://www.geeksforgeeks.org/>

*Stack Overflow en español*. (s. f.). Stack Overflow en español. <https://es.stackoverflow.com/>

Para obtener la base del script para hacer que las letras desaparecieran se usó:

Trellat. (2023, 26 octubre). Texto parpadeante con Javascript. Trellat. <https://trellat.es/texto-parpadeante-con-javascript/>

Para repasar las bases de los arrays multidimensionales se usó la siguiente página

Olawanle, J. (2023, 20 enero). JavaScript 2D Array – Two dimensional arrays in JS. freeCodeCamp.org. <https://www.freecodecamp.org/news/javascript-2d-arrays/>

Para visualizar todos los ASCII para representar los distintos sprites se usó:

Peter. (s. f.). The complete table of ASCII characters, codes, symbols and signs, American Standard Code for Information Interchange, The complete ASCII table, characters,letters, vowels with accents, consonants, signs, symbols, numbers ascii, ascii art, ascii table, code ascii, ascii character, ascii text, ascii chart, ascii characters, ascii codes, characters, codes, tables, symbols, list, alt, keys, keyboard, spelling, control, printable, extended, letters, epistles, handwriting, scripts, lettering, majuscules, capitals, minuscules, lower, case, small, acute, accent, sharp, engrave, diaresis, circumflex, tilde, cedilla, anillo, circlet, eñe, enie, arroba, pound, sterling, cent, type, write, spell, spanish, english, notebooks, laptops, ascii, asci, asccii, asqui, askii, aski, aschi, aschii,20240124. The complete table of ASCII characters, codes, symbols and signs. <https://theasciicode.com.ar/>

Para obtener y repasar el funcionamiento y propiedades del posicionamiento flex se usó:

*Flexbox Froggy*. (s. f.). A game for learning CSS flexbox. <https://flexboxfroggy.com/>

Como ayuda en la creación de sprites se utilizó el descubierto proyecto de github:

* <https://github.com/fr0zn/TreeDude>

Para ayuda en la documentación y explicación de conceptos se usaron las siguientes páginas:

Google. (s. f.). <https://www.google.com/>

colaboradores de Wikipedia. (s. f.). *Wikipedia, la Enciclopedia libre*. <https://es.wikipedia.org/>

School, T. (2023, 28 junio). Sprite videojuegos: ¿qué son y para qué sirven? Tokio School. <https://www.tokioschool.com/noticias/sprite-videojuegos/>

*Antes de continuar*. (s. f.). <https://translate.google.es/>

Machuca, F. (2022, 1 abril). *Aprende qué es un mapa de bits y haz que tus trabajos resalten por su calidad*. <https://www.crehana.com>.

<https://www.crehana.com/blog/estilo-vida/que-es-mapa-bits/>

# APÉNDICES

De dónde se sacó las ideas del proyecto

Save 70% on SUPERHOT on Steam. (s. f.). <https://store.steampowered.com/app/322500/SUPERHOT/>

*Typing Speed Test - Online typing test*. (s. f.). <https://typing-speed-test.aoeu.eu/>