GRADO EN DISEÑO Y DESARROLLO DE VIDEOJUEGOS
VJ1217: DISEÑO Y DESARROLLO DE JUEGOS WEB
Prueba de programación
5 de julio de 2022

Usuario examen: **usuario-de-examen**
Contraseña: **la-del-examen**

Nombre y apellidos: _____

DNI: **dni-de-examen**

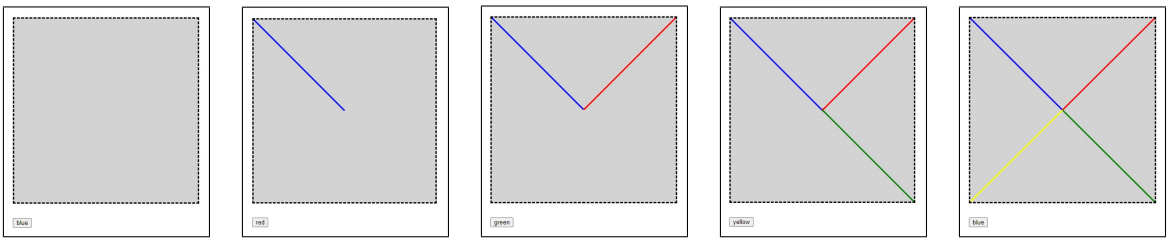
Previo. Descarga en el Escritorio de tu ordenador el paquete ExamenJulio.zip de *Aula Virtual*. Está en la sección **Exams**. Descomprímelo y desempaquétalo para obtener la carpeta ExamenJulio, que contiene tres subcarpetas dedicadas a cada una de las siguientes secciones.

ATENCIÓN: A la hora de probar el resultado de los diferentes ejercicios en el navegador debes emplear la opción Open with Live Server de VS Code tal como hemos visto en las clases de prácticas.

HTML5/CSS3 (2,5 ptos.)

Abre en VS Code la carpeta Bloque I - HTML-CSS. Al abrirla, encontrarás un fichero HTML denominado `index.html`. Realiza los cambios necesarios sobre ese fichero (no crees ningún fichero adicional) teniendo en cuenta las siguientes indicaciones y los resultados mostrados en las Figuras 1 y 2. Te resultará útil ver el vídeo que hemos dejado en *Aula Virtual* para que aprecies el funcionamiento que hay que conseguir al acabar los ejercicios de este bloque.

1. Añade una regla de estilo para que el canvas tenga como borde una línea discontinua y fondo de color gris claro. **Nota**: No añadas atributos al elemento correspondiente del canvas; usa una regla CSS.
2. Añade un botón según se muestra en la Figura 1 (esquina inferior izquierda, debajo del canvas) y asocia la función `drawLine()` a su evento de pulsación.
3. Define la función `draw()` para que dibuje una línea de color azul con un grosor (inicial) de 3 desde la posición $(0, 0)$ al centro del canvas.



Apartado 2 | Apartado 3 | Apartados 4 y 5 | Apartados 4 y 5 | Apartados 4 y 5

Figura 1: Resultados intermedios en el bloque de HTML5/CSS3

4. Adapta `draw()` para que dibuje líneas de colores distintos, según las siguientes consideraciones:

   a) El color de la línea es un parámetro de entrada de la función.

b) Según el color, el origen de la línea es distinto.

c) El punto final de la línea es siempre el centro del canvas.

5. Define `drawLine()` para que, en cada pulsación del botón, dibuje una línea de un color distinto según la secuencia definida en `COLORS`. La etiqueta del botón debe cambiar al siguiente color de la lista.

6. Adapta `drawLine()` para que cuando se alcance el último color de `COLORS` empiece otra vez con el primer color de la lista. Además, cada nuevo ciclo de colores el grosor de la línea aumentará en 2 unidades, tal como se muestra en la Figura 2. **Nota**: Modifica convenientemente `draw()`.

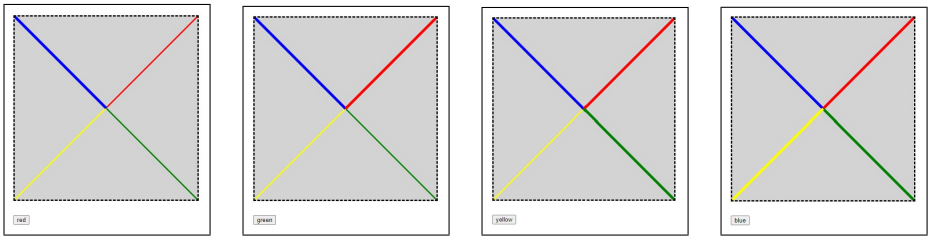
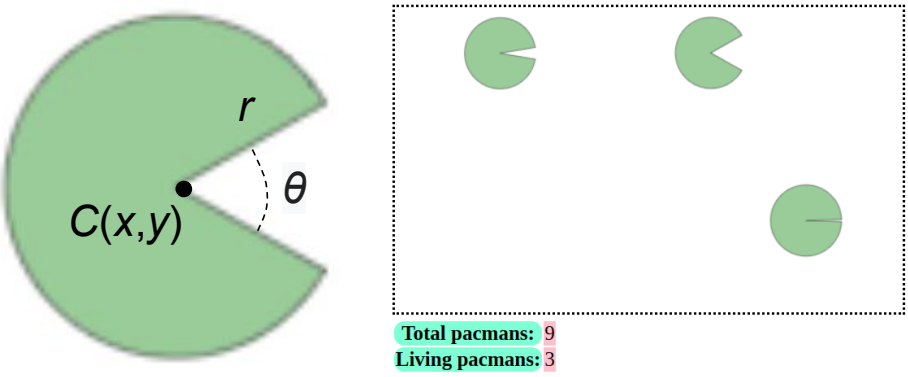


Figura 2: Resultados intermedios en el bloque de HTML5/CSS3

JavaScript (4,0 ptos.)

Abre en VS Code la carpeta Bloque II - JavaScript. Al abrirla, encontrarás un fichero HTML, denominado `index.html`, y una carpeta, js. Dentro de dicha carpeta hay un fichero llamado `pacman.js`.

Completa el fichero `pacman.js` con las siguientes indicaciones. Dicho fichero incluye valores en el objeto `game` que **deberás utilizar** adecuadamente en tu código. Se necesita consultar, pero **no se debe modificar**, el contenido de `index.html`.



(a) Pacman: $C(x, y)$, r y θ (b) Varios pacman en un canvas

Figura 3: Pacman: (a) parámetros que definen el estado de un pacman y (b) ejemplos de pacman dibujados en un canvas

7. Para representar y manipular *pacmans* (Figura 3a), completa la clase `Pacman` tal que:

a) El constructor reciba las coordenadas (x, y) de su posición C, y el radio r. Inicialmente, el ángulo de apertura θ es 0 (boca cerrada).

b) El método `move(shiftX,angleRange)` cambie la coordenada X del pacman en `shiftX` unidades, y θ se elija aleatoriamente en el intervalo continuo $[0, \texttt{angleRange})$, con `angleRange` dado en grados sexagesimales.

c) El método `draw(ctx)` dibuje el pacman usando el contexto `ctx` recibido como argumento. Mantén las líneas de código ya proporcionadas.

d) El método `isAboutToLeave(cv)` devuelva como valor lógico si el centro C del pacman se sale del canvas `cv` por la derecha.

8. Para crear, mover y dibujar pacmans, completa las siguientes funciones:

   a) `createPacman(cv)` que cree un pacman y lo añada a la propiedad `pacmans` de `game`. El centro del nuevo pacman se debe elegir aleatoriamente dentro del área del canvas `cv`, y su radio será 35 (recuerda usar la pertinente propiedad de `game`).
   b) `movePacmans()` que mueva todos los pacmans que hay en el array en `game`, cada uno según (7*b*). Tras mover cada pacman, debe comprobarse si está saliendo del canvas, usando el criterio de (7*d*) y, en caso afirmativo, debe eliminarse dicho pacman del array. Los argumentos con los que invocar a `move()` y `isAboutToLeave()` están en las correspondientes propiedades de `game`. No te preocupes por el valor inicial `null` del canvas de `game`; asume que tendrá un valor correcto cuando se ejecute `movePacmans()`.
   c) `cleanCanvas(cv)` para «limpiar» el canvas `cv`; es decir, eliminar todo lo que se haya dibujado.
   d) `drawPacmans(cv)` para limpiar el canvas `cv` y después dibujar todos los pacmans del array en `game`. Apóyate en `cleanCanvas()` de (8*c*).

9. Para saber y mostrar cuántos pacmans hay en cada momento y cuántos se han creado desde la carga de la página:

   a) Añade otra propiedad a `game` para saber cuántos pacman se han creado. Inicializa dicha propiedad a 0, e increméntala en una unidad en `createPacman()`.
   b) Escribe `displayCounts()` para mostrar dichos valores debajo del canvas, como en la Figura 3b.

10. Para gestionar pacmans (Figura 3b) completa la función `start()`:

    a) Inicializa la propiedad `canvas` de `game` con el canvas de la página cuyo identificador es pacman.
    b) Arranca un temporizador para crear un pacman cada 3 segundos.
    Para poder pasar el canvas a `createPacman()`, usa una función anónima. Por ejemplo, la función anónima

    ```
    function()  { f(3,  'Hi!'); }
    ```

    permitiría la llamada a la función `f()` con los argumentos `3` y `'Hi!'` de relevancia para esa hipotética `f()`.
    c) Inicia otro temporizador para mover y redibujar todos los pacmans siete veces por segundo. Con esa misma frecuencia, actualiza también los dos contadores de pacmans (Ejercicio 9). Identifica en el código la función ya definida que debes utilizar como *handler*.

Phaser ($3{,}5$ ptos.)

Abre en VS Code la carpeta Bloque III - Phaser. Al abrirla, encontrarás una estructura de carpetas y ficheros que conforman un juego similar al Shoot 'em up que estudiamos en la práctica 5 —es exactamente dicho juego, al que se le ha modificado muy ligeramente el primer estado y se le ha cambiado por completo el tercero, el que mostraba el *Hall of Fame* al final, por uno muy simple—.

Desarrolla los siguientes ejercicios independientes partiendo del código dado. Si tienes dudas acerca del funcionamiento de este juego, puedes consultar el vídeo que hay disponible en *Aula Virtual* y que muestra cómo quedaría el juego tras las modificaciones pedidas.

11. Sustituye la imagen de la nave del jugador por la animación del fichero assets/imgs/animacraft.png. Observa en este fichero que la imagen actual de la nave es el fotograma séptimo, y que los seis primeros fotogramas representan una animación de movimiento hacia la izquierda y los seis últimos, hacia la derecha. Todos los fotogramas son de 70×50 píxeles. En ambos casos, la animación de la nave se centra en la evolución de unos impulsores laterales que empujan a la nave desde el lado contrario al movimiento. Por supuesto, debes asociar adecuadamente la ejecución de cada animación a la pulsación de los cursores izquierdo y derecho y al movimiento del ratón, y el fotograma de parada cuando dejen de pulsarse o moverse, en el mismo sentido en que ocurría con la nave (imagen estática) en el juego original.
12. Incorpora sonidos al estado inicial. En concreto, haz que comience la audición en bucle del fichero assets/snds/intro.mp3 desde el mismo comienzo del estado inicial. Debe estar oyéndose hasta que se pulse el botón, momento en el que debe pararse e iniciarse la audición del otro fichero nuevo, assets/snds/toplay.mp3. Este también sonará en bucle hasta que la animación de salida de la nave de la escena se complete, y comience el juego, momento en el que debe pararse este segundo audio.
13. Incorpora tipos de letra en web y locales al juego. En concreto, prepara el código necesario para poder disponer de distintos tipos de letra en web y locales, y a continuación, asigna los siguientes tipos:

    a) Skranji, que es un tipo de letra en web de Google, debe asignarse a los créditos del estado inicial.
    b) TimeandSpaceRegular, que es un tipo de letra disponible en https://fontlibrary.org//face/time-and-space, debe asignarse a las instrucciones del estado inicial.
    c) Glitch Inside, que es un tipo de letra local disponible en el fichero assets/fonts/GlitchInside.otf, debe asignarse a un nuevo título que debes crear en el estado inicial:
        - Texto: Martian Tourists.
        - Tamaño: 64 píxeles.
        - Color: #CCCCCC (gris claro).
        - Ubicación: x=TEXT_OFFSET_HOR, y=(game.world.height-TEXT_OFFSET_VER)/2.
        - Anclaje: $x = 0.0$, $y = 0.5$.

14. En el estado inicial, presenta el título, recién introducido, carácter a carácter, a ritmo de uno por segundo. Desde su creación y mientras el título no esté completo, el botón debe estar inhabilitado aunque visible en la escena. Después, debe habilitarse para que pueda actuar tal cual lo hacía. Todos los demás elementos del estado inicial seguirán actuando igual que lo hacían, durante la construcción del título y después.

Entrega: Una vez hayas concluido, abre la carpeta Espai en disc personal que hay en el Escritorio de Windows y copia, dentro de ella, la carpeta ExamenJulio, que debe tener todos los contenidos de los ejercicios que has estado desarrollando. Si no copias esta carpeta en Espai en disc personal será como si no hubieses efectuado el examen.