Pol Caparrós Escudero - 1633937

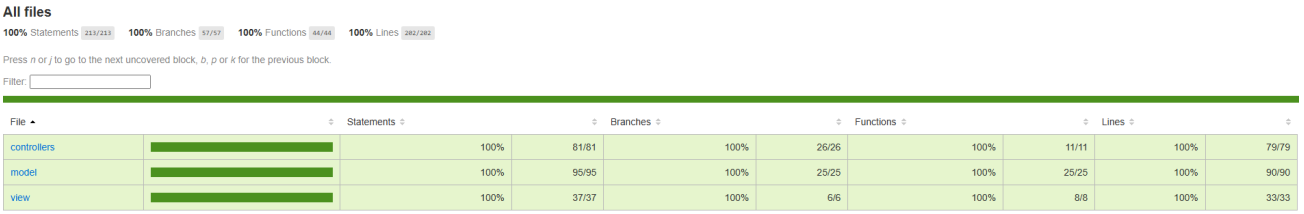
Marc Campo Cabrera - 1633330

Dijous 12:30

Arkanoid

Coverage total del programa:

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	100	100	100	100	
controllers	100	100	100	100	
GameController.ts	100	100	100	100	
model	100	100	100	100	
Ball.ts	100	100	100	100	
Brick.ts	100	100	100	100	
Map.ts	100	100	100	100	
Paddle.ts	100	100	100	100	
view	100	100	100	100	
GameView.ts	100	100	100	100	



Per poder revisar el coverage de manera més interactiva, podem fer:

- npm install -g pnpm (si no tenim pnpm)
- pnpm test
- Es generarà una carpeta /coverage/lcov-report on podem trobar un index.html, aquí podrem veure el coverage i navegar pel codi amb més informació.

En cas de voler executar el joc, executar

- pnpm run dev

Hem entés que hem de redactar la plantilla per cada funció, l'hem omplert el millor que hem pogut ja que l'hem trobat poc intuitiva per saber que es demanava a cada punt.

Hem realitzat TDD per a cada funció del codi, fent test - implementació fins completar la funció. Al final de tot hem creat els tests de Particions, Fronteres i Límits i els hem fet commit a la vegada ja que realment no s'ha canviat la implementació perquè ja funcionava.

Com podem observar al coverage, hem realitzar Statement, Decision i Path Coverage de tot el codi del programa. A les següents seccions indicarem on hem fet loop testing, condition coverage, Mocks, etc.

Com funciona Jest?

A diferència de JUnit, Jest inclou mocks automàtics, espies i generació d'informes de cobertura

de codi sense configuracions addicionals. En lloc d' anotacions com @Test, utilitza funcions com it() per definir proves. També executa proves en paral·lel per defecte, cosa que millora la velocitat.

Estrucura basica de Jest:

- **describe**: Agrupa proves relacionades, com un context o funcionalitat específica.
- **it**: Defineix un cas de prova individual, similar a @Test a JUnit.
- **beforeEach**: Executa codi abans de cada prova en el bloc on es defineix, útil per inicialitzar estats o variables.
- **afterEach**: Executa codi després de cada prova en el bloc on es defineix, útil per destruir estats o variables.
- **expect**: equivalent a assert.

CLASSE GAME VIEW

Funcionalitat:

Configuració inicial del canvas perquè tingui les dimensions adequades per a la visualització del joc.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: loadCanvas()

Test:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("debería inicializar el canvas con el tamaño correcto").

Tipus de test: Caixa blanca.

Tècniques utilitzades: Statement coverage, verificació que els atributs width i height del canvas es configuren correctament.

Funcionalitat:

Renderització de la bola, el paddle i el mapa de blocs.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: render(ball: Ball, paddle: Paddle, map: Map)

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("debería llamar a drawBall cuando render es llamado")
- it("debería llamar a drawBall, drawPaddle y drawMap cuando render es llamado")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Mockups i espies (spyOn) per verificar que els mètodes interns es criden correctament.

Funcionalitat:

Actualització del marcador amb un nou valor.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: updateScore(score: number)

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("debería actualizar el contenido del scoreDiv con el puntaje proporcionado")
- it("debería actualizar el contenido del scoreDiv con un puntaje negativo")
- it("debería actualizar el contenido del scoreDiv con el puntaje de 0")

Tipus de test: Caixa negra.

Tècniques utilitzades: Verificació de l'HTML generat amb valors positius, negatius i zero.

Funcionalitat:

Dibuix de la bola al canvas.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: drawBall(ball: Ball)

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("Debería dibujar la bola dentro del canvas")
- it("Debería dibujar la bola parcialmente fuera del canvas")
- it("Debería dibujar la bola completamente fuera del canvas")
- it("Debería dibujar la bola en la frontera del borde izquierdo")
- it("Debería dibujar la bola en la frontera del borde derecho")
- it("Debería dibujar la bola en el límite inferior del borde izquierdo")
- it("Debería dibujar la bola en el límite superior del borde izquierdo")
- it("Debería dibujar la bola en el límite inferior del borde derecho")
- it("Debería dibujar la bola en el límite superior del borde derecho")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Particions equivalents per verificar posicions dins i fora del canvas, i verificació de fronteres i límits per assegurar comportament correcte a les vores.

Funcionalitat:

Dibuix del paddle al canvas.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: drawPaddle(paddle: Paddle)

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("Debería dibujar el paddle completamente dentro del canvas")
- it("Debería dibujar el paddle parcialmente fuera del canvas")
- it("Debería dibujar el paddle completamente fuera del canvas")
- it("Debería dibujar el paddle en la frontera izquierda del canvas")
- it("Debería dibujar el paddle en la frontera derecha del canvas")
- it("Debería dibujar el paddle fuera del canvas (límite inferior izquierdo)")
- it("Debería dibujar el paddle dentro del canvas (límite superior izquierdo)")
- it("Debería dibujar el paddle dentro del canvas (límite inferior derecho)")
- it("Debería dibujar el paddle fuera del canvas (límite superior derecho)")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Particions equivalents i verificació de fronteres per assegurar comportament correcte a les vores del canvas.

Funcionalitat:

Dibuix del mapa de blocs al canvas, només representant els blocs amb estat ALIVE.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: drawMap(map: Map)

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("debería dibujar solo los ladrillos con estado ALIVE")
- it("no debería dibujar ladrillos con estado DEAD")
- it("debería manejar un array vacio de ladrillos correctamente")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Mockups per simular el mapa (Map) i verificar que només es criden els dibuixos per als blocs ALIVE.

Funcionalitat:

Neteja completa del canvas per eliminar contingut anterior.

Localització:

Arxiu: GameView.ts

Classe: GameView

Mètode desenvolupat: clearCanvas()

Tests:

Arxiu: GameView.test.ts

Classe: GameView

Tests associats:

- it("debería limpiar el canvas")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Mockups per verificar que clearRect es crida amb els arguments correctes.

CLASSE PADDLE

Funcionalitat:

Comprovació de col·lisions del paddle amb els límits esquerre i dret del canvas. Assigna els valors de collisionLeft i collisionRight correctament.

Localització:

Arxiu: Paddle.ts

Classe: Paddle

Mètode desenvolupat: checkCollision()

Tests:

Arxiu: Paddle.test.ts

Classe: Paddle

Tests associats:

- it("debería ejecutar checkCollision sin errores")
- it("debería llamar a checkCollisionCanvasRight y checkCollisionCanvasLeft")
- it("debería assignar collisionRight y collisionLeft de manera correcta")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Mockups per espiar les crides als mètodes checkCollisionCanvasRight i checkCollisionCanvasLeft, i verificació d'assignació de propietats.

Funcionalitat:

Verifica si el paddle pot moure's cap a la dreta, tenint en compte els límits del canvas.

Localització:

Arxiu: Paddle.ts

Classe: Paddle

Mètode desenvolupat: checkCollisionCanvasRight()

Tests:

Arxiu: Paddle.test.ts

Classe: Paddle

Tests associats:

- it("debería detectar que no hay colisión en el borde derecho cuando el paddle está lejos del borde del canvas")
- it("debería detectar colisión en el borde derecho cuando el paddle está al borde del canvas")
- it("debería detectar que no hay colisión en el borde derecho para una posición intermedia dentro del canvas")
- it("debería detectar colisión cuando la pala está tocando el borde derecho")
- it("debería permitir movimiento cuando la pala está completamente dentro del borde derecho")
- it("debería detectar colisión cuando la pala está más allá del borde derecho")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Verificació de valors frontera i particions equivalents per assegurar que el comportament és correcte en posicions intermitges i en els límits.

Funcionalitat:

Verifica si el paddle pot moure's cap a l'esquerra, tenint en compte els límits del canvas.

Localització:

Arxiu: Paddle.ts

Classe: Paddle

Mètode desenvolupat: checkCollisionCanvasLeft()

Tests:

Arxiu: Paddle.test.ts

Classe: Paddle

Tests associats:

- it("debería detectar que no hay colisión en el borde izquierdo cuando el paddle está lejos del borde izquierdo")
- it("debería detectar colisión en el borde izquierdo cuando el paddle está al borde del canvas")
- it("debería detectar que no hay colisión en el borde izquierdo para una posición intermedia dentro del canvas")
- it("debería detectar colisión cuando la pala está tocando el borde izquierdo")
- it("debería permitir movimiento cuando la pala está completamente dentro del borde izquierdo")
- it("debería detectar colisión cuando la pala está más allá del borde izquierdo")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Verificació de valors frontera i particions equivalents per assegurar que el comportament és correcte en posicions intermitges i en els límits.

Funcionalitat:

Permet moure el paddle cap a l'esquerra o la dreta en funció de l'estat de les col·lisions (collisionLeft, collisionRight) i de les tecles pressionades (rightPressed, leftPressed).

Localització:

Arxiu: Paddle.ts

Classe: Paddle

Mètode desenvolupat: move(rightPressed: boolean, leftPressed: boolean)

Tests:

Arxiu: Paddle.test.ts

Classe: Paddle

Tests associats:

- it("debería mover el paddle a la derecha cuando rightPressed es true y collisionRight es true")
- it("debería mover el paddle a la izquierda cuando leftPressed es true y collisionLeft es true")
- it("no debería mover el paddle si no hay colisión en la respectiva dirección")
- it("debería moverse solo en la dirección permitida por la colisión cuando ambas teclas están presionadas")
- it("debería detectar movimiento correcto en los valores límite")
- it("no debería moverse más allá de los valores frontera")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Condition coverage per verificar totes les combinacions de col·lisions i moviments. També es van utilitzar particions equivalents i valors límit per assegurar el comportament en els extrems.

Funcionalitat:

Inicialitza les propietats del paddle, com les dimensions (paddleWidth, paddleHeight), la posició (paddleX, paddleY), i altres valors inicials.

Localització:

Arxiu: Paddle.ts

Classe: Paddle

Mètode desenvolupat: constructor(paddleWidth, paddleHeight, paddleX, paddleY)

Tests:

Arxiu: Paddle.test.ts

Classe: Paddle

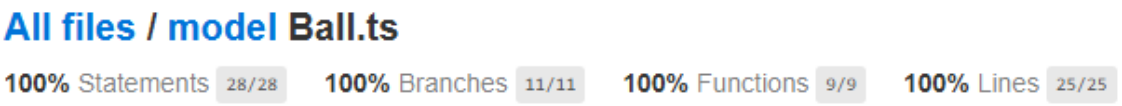
Tests associats:

- it("debería asignar el ancho de la pala (paddleWidth) correctamente")
- it("debería asignar la altura de la pala (paddleHeight) correctamente")
- it("debería asignar la posición X de la pala (paddleX) correctamente")
- it("debería asignar la posición Y de la pala (paddleY) correctamente")
- it("debería inicializar correctamente con dimensiones válidas")
- it("debería manejar dimensiones inválidas (negativas)")
- it("debería manejar posiciones iniciales fuera del rango del canvas")

Tipus de test: Caixa negra.

Tècniques utilitzades: Particions equivalents per verificar dimensions vàlides i vàlors fora de rang, així com verificació de valors frontera.

CLASSE BALL



Funcionalitat:

Inicialitza les propietats de la bola, com la posició (x, y), el radi (radius) i les velocitats (speedX, speedY).

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: constructor(x: number, y: number, radius: number, speedX: number, speedY: number)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("debería inicializar las propiedades correctamente")

Tipus de test: Caixa negra.

Tècniques utilitzades: Particions equivalents per verificar que es poden inicialitzar valors vàlids i que es guarden correctament a les propietats de la classe.

Funcionalitat:

Inverteix la velocitat horitzontal (speedX) i actualitza la posició x.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: changeX()

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("debería cambiar la dirección de speedX y actualizar la posición X")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Statement coverage per assegurar que s'actualitza tant la velocitat com la posició.

Funcionalitat:

Inverteix la velocitat vertical (speedY) i actualitza la posició y.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: changeY()

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("debería cambiar la dirección de speedY y actualizar la posición Y")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Statement coverage per assegurar que s'actualitza tant la velocitat com la posició.

Funcionalitat:

Actualitza la posició de la bola tenint en compte les col·lisions amb la pala i les parets del canvas.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: move()

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("debería actualizar la posición de la bola según speedX y speedY")
- it("debería invertir speedY al colisionar con la pala")
- it("debería invertir speedX al colisionar con el canvas X")
- it("debería invertir speedY al colisionar con el canvas Y")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Verificació de condicions per assegurar que les col·lisions modifiquen adequadament les velocitats i que la posició es calcula correctament.

Funcionalitat:

Detecta si la bola ha sortit del límit inferior del canvas.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: ballDownMap(canvasHeight: number)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("debería detectar colisión en el borde inferior del canvas")
- it("no debería detectar colisión cuando está dentro del canvas")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Particions equivalents per comprovar quan la bola està dins o fora del canvas.

Funcionalitat:

Detecta si la bola ha col·lisionat amb la pala.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: checkCollisionPaddle(paddleX, paddleY, paddleWidth, paddleHeight)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("Debería detectar una colisión cuando el objeto está en el rango de x e y del paddle")
- it("No debería detectar colisión cuando el objeto no está en el rango de y del paddle")
- it("No debería detectar colisión cuando el objeto no está en el rango de x del paddle")
- it("Debería detectar colisión en la frontera del borde derecho")
- it("Debería detectar colisión en el límite superior del borde derecho")
- it("Debería detectar colisión en la frontera del borde izquierdo")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Verificació de valors frontera i particions equivalents per assegurar deteccions correctes dins i fora del rang.

Funcionalitat:

Detecta si la bola ha col·lisionat amb els límits esquerre o dret del canvas.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: checkCollisionCanvasX(canvasWidth: number)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("Dería detectar una colisión en el borde izquierdo del canvas")
- it("Debería detectar una colisión en el borde derecho del canvas")
- it("No debería detectar colisión cuando el objeto está dentro del canvas")
- it("Debería detectar colisión en el límite inferior del borde izquierdo")
- it("No debería detectar colisión en el límite superior del borde izquierdo")
- it("Debería detectar colisión en el límite superior del borde derecho")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Particions equivalents i verificació de fronteres per assegurar deteccions correctes dins i fora dels límits.

Funcionalitat:

Detecta si la bola ha col·lisionat amb els límits superior o inferior del canvas.

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: checkCollisionCanvasY(canvasHeight: number)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

Tests associats:

- it("Debería detectar una colisión en el borde superior del canvas")
- it("Debería detectar una colisión en el borde inferior del canvas")
- it("No debería detectar colisión cuando el objeto está dentro del canvas")
- it("Debería detectar colisión en la frontera del borde superior")
- it("No debería detectar colisión en el límite superior del borde superior")
- it("Debería detectar colisión en el límite inferior del borde inferior")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Particions equivalents i verificació de fronteres per assegurar deteccions correctes dins i fora dels límits.

Funcionalitat:

Detecta si la bola ha col·lisionat amb la pala o amb els límits del canvas (tant horitzontals com verticals).

Localització:

Arxiu: Ball.ts

Classe: Ball

Mètode desenvolupat: checkCollision(paddleX, paddleY, paddleWidth, paddleHeight, canvasWidth, canvasHeight)

Tests:

Arxiu: Ball.test.ts

Classe: Ball

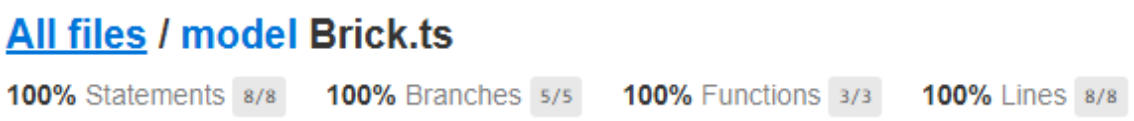
Tests associats:

- it("debería detectar colisión con la pala")
- it("debería detectar colisión con los bordes izquierdo y derecho del canvas (eje X)")
- it("debería detectar colisión con los bordes superior e inferior del canvas (eje Y)")
- it("no debería detectar colisión cuando la bola está lejos de la pala y de los bordes del canvas")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Mockups per simular col·lisions amb diferents elements i assegurar que els estats de col·lisió es gestionen correctament.

CLASSE BRICK



Funcionalitat:

Inicialitza les propietats del Brick (BrickX, BrickY, status, color) amb els valors proporcionats.

Localització:

Arxiu: Brick.ts

Classe: Brick

Mètode desenvolupat: constructor(BrickX: number, BrickY: number, status: number, color: number)

Tests:

Arxiu: Brick.test.ts

Classe: Brick

Tests associats:

- it("debería inicializarse con las propiedades correctas")
- it("debería inicializarse con el estado ALIVE cuando el estado es 1")
- it("debería inicializarse con el estado DEAD cuando el estado es 0")
- it("debería inicializar correctamente con valores válidos")
- it("debería manejar correctamente valores frontera para coordenadas")
- it("debería manejar valores límite fuera del rango esperado")
- it("debería manejar valores frontera para color")

Tipus de test: Caixa negra.

Tècniques utilitzades:

Particions equivalents per assegurar la inicialització de valors vàlids i verificació de valors frontera per coordenades i colors.

Funcionalitat:

Canvia l'estat del Brick de ALIVE a DEAD.

Localització:

Arxiu: Brick.ts

Classe: Brick

Mètode desenvolupat: hit()

Tests:

Arxiu: Brick.test.ts

Classe: Brick

Tests associats:

- it("debería llamar a la clase hit() y actualizar el status del brick")
- it("debería cambiar el estado del ladrillo a DEAD si está ALIVE")
- it("debería manejar correctamente si el ladrillo ya está DEAD")

Tipus de test: Caixa blanca.

Tècniques utilitzades: Statement coverage per assegurar el canvi correcte de l'estat del Brick.

Funcionalitat:

Detecta si una bola col·lisiona amb el Bricktenint en compte les dimensions i la posició del Brick, així com el seu estat.

Localització:

Arxiu: Brick.ts

Classe: Brick

Mètode desenvolupat: isHit(ballX: number, ballY: number, brickWidth: number, brickHeight: number)

Tests:

Arxiu: Brick.test.ts

Classe: Brick

Tests associats:

Particions equivalents:

- it("debería devolver true si la bola golpea el ladrillo")
- it("debería devolver false si la bola está fuera del rango en X")
- it("debería devolver false si la bola está fuera del rango en Y")
- it("debería devolver false si el ladrillo está muerto")
- it("debería devolver true si la bola está completamente dentro del ladrillo")
- it("debería devolver false si la bola está completamente fuera del rango del ladrillo")

Límits i fronteres:

- it("debería detectar colisión justo en la frontera izquierda")
- it("debería devolver false justo en el límite inferior izquierdo")
- it("debería detectar colisión justo dentro del límite superior izquierdo")
- it("debería detectar colisión justo en la frontera derecha")
- it("debería detectar colisión justo dentro del límite inferior derecho")
- it("debería devolver false justo en el límite superior derecho")
- it("debería detectar colisión justo en la frontera superior")
- it("debería devolver false justo en el límite inferior superior")
- it("debería detectar colisión justo dentro del límite superior superior")
- it("debería detectar colisión justo en la frontera inferior")
- it("debería detectar colisión justo dentro del límite inferior inferior")
- it("debería devolver false justo en el límite superior inferior")

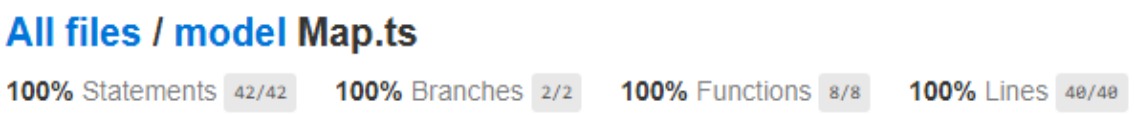
Tipus de test: Caixa blanca.

Tècniques utilitzades:

Particions equivalents per provar l'estat del Brick (ALIVE vs. DEAD), la posició de la bola (dins vs. fora del rang), i les col·lisions generals.

Verificació de fronteres i límits per assegurar deteccions correctes just als marges del Brick.

CLASSE MAP



Funcionalitat:

Retorna l'array de Bricks (bricks) generat.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: getBricks()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería devolver el array de bricks con getBricks")
- it("debería devolver valores personalizados asignados a las propiedades")

Tipus de test: Caixa negra.

Tècniques utilitzades: Verificació de particions equivalents per assegurar que el retorn inicial i personalitzat són correctes.

Funcionalitat:

Retorna el nombre de columnes de Bricks.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: getBrickColumnCount()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería devolver el número de columnas de ladrillos con getBrickColumnCount")
- it("debería devolver valores personalizados asignados a las propiedades")
- it("debería manejar dimensiones de matriz en la frontera y límites")

Tipus de test: Caixa negra.

Tècniques utilitzades: Particions equivalents i valors límit per verificar valors predeterminats i personalitzats.

Funcionalitat:

Retorna el nombre de files de Bricks.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: getBrickRowCount()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería devolver el número de filas de ladrillos con getBrickRowCount")
- it("debería devolver valores personalizados asignados a las propiedades")
- it("debería manejar dimensiones de matriz en la frontera y límites")

Tipus de test: Caixa negra.

Tècniques utilitzades: Particions equivalents i valors límit per assegurar un comportament correcte.

Funcionalitat:

Retorna l'ample dels Bricks.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: getBrickWidth()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería devolver el ancho de los ladrillos con getBrickWidth")
- it("debería devolver valores personalizados asignados a las propiedades")
- it("debería manejar dimensiones de ladrillos en la frontera y límites")

Tipus de test: Caixa negra.

Tècniques utilitzades:

Particions equivalents i valors frontera.

Funcionalitat:

Retorna l'altura dels Bricks.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: getBrickHeigth()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería devolver la altura de los ladrillos con getBrickHeigth")
- it("debería devolver valores personalizados asignados a las propiedades")
- it("debería manejar dimensiones de ladrillos en la frontera y límites")

Tipus de test: Caixa negra.

Tècniques utilitzades:

Particions equivalents i valors frontera.

Funcionalitat:

Assigna valors a les propietats del mapa en funció del nivell seleccionat i genera els Bricks corresponents.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: selectLevel(level: number)

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería asignar valores correctos al seleccionar el nivel 1")
- it("debería reiniciar las propiedades al seleccionar un nivel no válido")
- it("debería llamar a generateBricks al seleccionar cualquier nivel")
- it("debería manejar niveles válidos y no válidos en la frontera y límites")

Tipus de test: Caixa blanca.

Tècniques utilitzades:

Statement coverage per verificar l'execució de tots els camins lògics.

Particions equivalents per nivells vàlids i no vàlids.

Límits i frontera per assegurar un comportament correcte als marges.

Funcionalitat:

Genera els Bricks del mapa basant-se en les dimensions, el desplaçament i l'espaiat configurats.

Localització:

Arxiu: Map.ts

Classe: Map

Mètode desenvolupat: generateBricks()

Tests:

Arxiu: Map.test.ts

Classe: Map

Tests associats:

- it("debería inicializar la matriz de bricks con las dimensiones correctas")
- it("debería asignar posiciones correctas a los bricks")
- it("debería inicializar cada ladrillo como una instancia de Brick")
- it("debería testear el bucle interno (r = 0,1,2,4,6,7) con c = 1")
- it("debería testear el bucle externo (c = 0,1,2,8,12,13) con r = 7")
- it("debería manejar valores frontera y límites para ladrillos")
- it("debería generar correctamente una matriz en la frontera y límites")

Tipus de test: Caixa blanca.

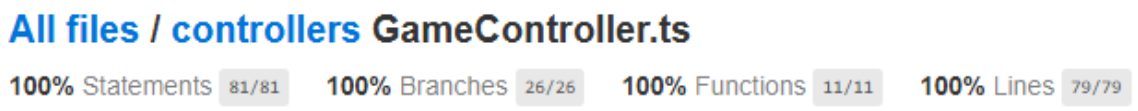
Tècniques utilitzades:

Loop testing per verificar el funcionament dels bucles interns i externs.

Statement coverage per assegurar la correcta inicialització de tots els Bricks.

Límits i frontera per dimensions i posicions dels Bricks .

CLASSE GAMECONTROLLER



Funcionalitat:

Inicia el joc, inicialitza els objectes principals i comença el bucle del joc.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: startGameMethod()

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería inicializar el GameController")
- it("debería iniciar el juego y comenzar el bucle del juego")
- it("debería inicializar la puntuación correctamente")
- it("debería inicializar las propiedades correctamente")

Tipus de test: Caixa negra i caixa blanca.

Tècniques utilitzades:

Mockups per inicialitzar dependències com Ball, Paddle i Map.

Statement coverage per assegurar la correcta execució de tots els passos del mètode, incloent la crida al bucle principal (gameLoop).

Funcionalitat:

Executa el bucle principal del joc per actualitzar i renderitzar l'estat actual del joc.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: gameLoop()

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería llamar a clearCanvas cuando gameLoop es ejecutado")
- it("debería llamar a render con ball, paddle y map cuando gameLoop es ejecutado")
- it("no debería llamar a clearCanvas ni render si startGame es false")
- it("debería llamar a checkCollisions, ballMove y paddleMove si game está en ejecución")
- describe("Pairwise Testing de gameLoop")
    - Dins hi ha un bucle per fer diferents tests del Pairwise

```
// PAIRWISE TESTING
describe("Pairwise Testing de gameLoop", () => {
  // Casos de prueba
  const testCases = [
    { startGame: true, isRunning: true },
    { startGame: true, isRunning: false },
    { startGame: false, isRunning: true },
    { startGame: false, isRunning: false },
  ];

  testCases.forEach(({ startGame, isRunning }) => {
    // Crear un caso de prueba para cada combinación de valores
    it(`debería manejar startGame=${startGame} y isRunning=${isRunning}`, () => {
      controller["startGame"] = startGame; // Configurar startGame
      controller["isRunning"] = isRunning; // Configurar isRunning

      controller["gameLoop"](); // Llamar al método

      if (startGame && isRunning) {
        // Si startGame y isRunning son true
        expect(mockView.clearCanvas).toHaveBeenCalled(); // Se llama a clearCanvas
        expect(mockView.render).toHaveBeenCalledWith(
          // Se llama a render con los objetos correctos
          controller["ball"],
          controller["paddle"],
          controller["map"]
        );
      } else {
        // Si startGame o isRunning son false
        expect(mockView.clearCanvas).not.toHaveBeenCalled(); // No se llama a clearCanvas
        expect(mockView.render).not.toHaveBeenCalled(); // No se llama a render
      }
    });
  });
});
```

Tipus de test: Caixa blanca i caixa negra.

Tècniques utilitzades:

Statement coverage per assegurar l'execució correcta de totes les parts del bucle.

Pairwise testing per provar combinacions de valors per a startGame i isRunning.

Mock fet manualment per mockejar el Paddle que es pasa a la vista en el render del gameLoop

Funcionalitat:

Detecta i maneja les col·lisions entre la bola, la paleta i els Bricks.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: checkCollisions()

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería finalizar el juego y recargar la página si la bola se sale del mapa")
- it("no debería finalizar el juego ni recargar la página si la bola no se sale del mapa")
- it("debería verificar la colisión en la Ball")
- it("debería verificar la colisión en el Paddle")
- it("debería llamar a mapCollision()")

Tipus de test: Caixa blanca.

Tècniques utilitzades:

Mockups per simular col·lisions amb objectes com la bola i la paleta.

Statement coverage per assegurar que totes les condicions de col·lisió són verificades.

Funcionalitat:

Detecta i processa col·lisions de la bola amb els Bricks del mapa, actualitza la puntuació i el moviment de la bola.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: mapCollision()

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería actualizar la puntuación cuando un ladrillo es golpeado")
- it("debería llamar a hit en el ladrillo cuando la bola lo golpea")
- it("no debería actualizar la puntuación si la bola no golpea el ladrillo")
- it("debería llamar a changeY cuando un ladrillo es golpeado")
- it("no debería llamar a changeY si la bola no golpea el ladrillo")

Tipus de test: Caixa blanca.

Tècniques utilitzades:

Mockups per simular el comportament dels Bricks.

Statement coverage per verificar que cada Brick és processat correctament.

Funcionalitat:

Gestiona les accions del jugador quan es premen tecles específiques (ArrowRight, ArrowLeft, Space) per iniciar el moviment o el joc.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: keyDownHandler(event: KeyboardEvent)

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería establecer rightPressed en true cuando se presiona 'ArrowRight'")
- it("debería establecer leftPressed en true cuando se presiona 'ArrowLeft'")
- it("debería iniciar el juego si se presiona la barra espaciadora cuando startGame es true y isRunning es false")
- it("debería detener el juego si se presiona la barra espaciadora cuando isRunning es true")

- it("no debería cambiar propiedades para teclas irrelevantes")

Tipus de test: Caixa negra i Caixa Blanca.

Tècniques utilitzades.

Particions equivalents: Per verificar el comportament esperat per a tecles rellevants (ArrowRight, ArrowLeft, Space) i no rellevants.

Condition coverage: Per assegurar que totes les condicions (tecles premudes) són verificades.

Mockups: Per simular els esdeveniments de teclat i comprovar l'efecte en les propietats del controlador.

Funcionalitat:

Gestiona les accions del jugador quan es deixen anar tecles específiques (ArrowRight, ArrowLeft) per aturar el moviment.

Localització:

Arxiu: GameController.ts

Classe: GameController

Mètode desenvolupat: keyUpHandler(event: KeyboardEvent)

Tests:

Arxiu: GameController.spec.ts

Classe: GameController

Tests associats:

- it("debería establecer rightPressed en false cuando se suelta 'ArrowRight'")
- it("no debería establecer rightPressed en false cuando no es 'ArrowRight'")
- it("debería establecer rightPressed en false cuando se suelta 'Right'")
- it("no debería establecer rightPressed en false cuando no es 'Right'")
- it("debería establecer leftPressed en false cuando se suelta 'ArrowLeft'")
- it("no debería establecer leftPressed en false cuando no es 'ArrowLeft'")
- it("debería establecer leftPressed en false cuando se suelta 'Left'")
- it("no debería establecer leftPressed en false cuando no es 'Left'")
- it("no debería cambiar propiedades para teclas irrelevantes")

Tipus de test: Caixa negra i Blanca.

Tècniques utilitzades:

Particions equivalents: Per verificar el comportament esperat per a tecles rellevants (ArrowRight, ArrowLeft) i no rellevants.

Condition coverage: Per assegurar que totes les condicions (tecles soltades) són verificades.

Mockups: Per simular els esdeveniments de teclat i comprovar l'efecte en les propietats del controlador.