

**17 de noviembre de 2025**

**Estructuras de Datos**

**Prof. José Calvo Suarez**

**Universidad Nacional de Costa Rica**

**Documento bitácora de IA**

**Arianna Barquero Ortiz**

**Sebastián Castro Guevara**

**Yoel Hernández Zamora**

**UNA**

la dificultad "fácil" no funciona, el NPC se queda quieto

[Juego.py](#)

GitHub Copilot

Step-by-step solution:

- The NPC logic uses a helper named `movimiento_aleatorio_npc()` in Fácil mode, but that method is missing, causing an exception and leaving the NPC idle.
- Implement `movimiento_aleatorio_npc()` to choose a valid neighboring cell (not a building) at random, or stay in place if none are available.
- No other logic changes are needed; the existing Fácil branch already calls this method and will begin moving once it exists.

**Juego.py**

Add the missing `movimiento_aleatorio_npc()` helper used by Fácil mode so the NPC moves randomly on the map.

[Juego.py +11 -0](#)

↻ ↺ ↻ ↻

```
def movimiento_aleatorio_npc(self):
    """
    Devuelve una celda vecina válida (no edificio) al azar para el NPC.
    Si no hay vecinos válidos, permanece en su celda actual.
    """

    cur = (self.npc_spriteRow, self.npc_spriteCol)
    vecinos = self._npc_neighbors(cur)
    if not vecinos:
        return cur
    return random.choice(vecinos)
```

podrás hacer una pantalla de selección de dificultad de NPC después de la pantalla del nombre; igualmente, no necesito que haga nada por el momento

[Juego.py](#)

Solución (pasos)

- Añadir variables de estado para el popup de dificultad (opciones, índice seleccionado, flag activo, resultado).
- Mostrar el popup de dificultad inmediatamente después de que el jugador confirme su nombre.
- Dibujar el popup y permitir selección con flechas / 1-3 / Enter / Esc.
- Guardar la opción seleccionada en `self.npc_difficulty` y actualizar el nombre del NPC para verificar visualmente.

c:\Users\XPC\Desktop\CourierQuest\juego.py

Añadir UI de selección de dificultad del NPC y su manejo de teclas.

[Juego.py +40 -1](#)

```
def draw_difficulty_popup(self):
    # ... (resto del código)

    self.difficulty_popup_active = False
    if self.difficulty_popup_active:
        self.difficulty_options = ["Fácil", "Normal", "Difícil"]
        self.difficulty_selected_idx = 1 # por defecto "Normal"
        self.npc_difficulty = None

    # ... (resto del código)
```

puedes arreglar la rotación del NPC al ir hacia la izquierda y luego arriba/abajo?

[Juego.py](#)

**Juego.py**

Para arreglar la rotación del NPC al moverse hacia la izquierda y luego arriba/abajo, debes asegurarte de que el ángulo y la escala\_x del sprite se actualicen correctamente en función de la dirección del movimiento, y que no se mantenga la escala\_x negativa al cambiar de dirección vertical.

**Solución paso a paso:**

- En la función `actualizar_npc`, cuando calculas el movimiento del NPC (`drx, dcy`), ajusta la escala\_x y el ángulo solo cuando el movimiento es horizontal. Cuando el movimiento es vertical, asegúrate de restaurar la escala\_x a positiva (mirando hacia arriba/abajo).
- Haz lo mismo en cualquier otro lugar donde se actualice la dirección del NPC.

**Cambios por archivo:**

[Juego.py +8 -4](#)

Con esto, el NPC rotará correctamente al cambiar de dirección, restaurando la escala\_x positiva al moverse verticalmente.

↻ ↺ ↻ ↻

```
if dr == -1:
    self.npc_sprite.angle = 270
    self.npc_sprite.scale_x = abs(self.npc_sprite.scale_x) # Mirar arriba, escala positiva
elif dr == 1:
    self.npc_sprite.angle = 90
    self.npc_sprite.scale_x = abs(self.npc_sprite.scale_x) # Mirar abajo, escala positiva
elif dc == -1:
    self.npc_sprite.angle = 0
    self.npc_sprite.scale_x = -abs(self.npc_sprite.scale_x) # Mirar Izquierda
elif dc == 1:
    self.npc_sprite.angle = 0
    self.npc_sprite.scale_x = abs(self.npc_sprite.scale_x) # Mirar derecha
```

puedes ver por qué no funciona el npc?

[Juego.py](#) [Main.py](#)

Solución (pasos)

- Inicializar correctamente la posición lógica del NPC (`row/col/peso`) y asegurar que tenga Resistencia y atributos usados.
- Cuando el NPC elige un pedido, que lo acepte (mover de pending a active) y se crean los sprites pickup/dropoff — si no, nunca encontrará sprites para recoger/entregar.
- Reemplazar la llamada a un método inexistente `self.npc_sprite.tiene_pedido(...)` por un chequeo seguro usando una función helper que ya existe (`npc_tiene_pedido`).
- Añadir pequeños ajustes defensivos (atributos inicializados) y registro de errores mínimo.

Cambios por archivo:

c:\Users\XPC\Desktop\CourierQuest\juego.py

- Inicializa `npc_sprite.row/col/peso_total`.
- Al asignar `npc_pedido_activo`, mueve el pedido a `pedidos_activos` y crea sus sprites (aceptación automática).
- Usa `npc_tiene_pedido(...)` en lugar de llamar a un método que posiblemente no existe en Repartidor.
- Añade protección ante excepciones

```
# ... (resto del código)

for i, opt in enumerate(self.difficulty_options):
    if opt == self.difficulty_selected_idx:
        bg = arcade.color.DARK_BLUE
        RBL = 1
    else:
        bg = arcade.color.DARK_GRAY
        RBL = 0.5
    self.difficulty_rectangles.append(arcade.draw_rectangle_outline(i * 100, 200, 100, 50, bg, 4, RBL))

arcade.draw_text("Fácil", 150, 200, arcade.color.DARK_GRAY, 16, anchor_x="left", anchor_y="center")
arcade.draw_text("Normal", 300, 200, arcade.color.DARK_GRAY, 16, anchor_x="left", anchor_y="center")
arcade.draw_text("Difícil", 450, 200, arcade.color.DARK_GRAY, 16, anchor_x="left", anchor_y="center")
```

puedes añadir otro jugador? no necesito que se mueva ni nada por el momento, solo ver si es posible.

[Juego.py](#)

Solución (pasos)

- Crear un nuevo sprite para el segundo jugador (NPC) en el inicializador (`init`), colocándolo en una celda válida distinta del jugador.
- Guardar el NPC en un `SpriteList` nuevo (`self.npc_list`).
- Dibujar el npc en `on_draw` y opcionalmente mostrar su nombre encima.
- No se añade lógica de movimiento: el NPC será estático.

c:\Users\XPC\Desktop\CourierQuest\juego.py

Añadir un NPC estático (segundo jugador) y dibujarlo.

[Juego.py +44 -29](#)

puedes hacer que si se escoge la dificultad 1, el npc se mueva aleatoriamente

[Juego.py:1759-1760](#)

[MapaPrueba.py](#)

Solución (pasos)

- Añadir método `movimiento_aleatorio_npc` que devuelve una celda adyacente válida aleatoria.
- Modificar `actualizar_npc` para usar `movimiento aleatorio` si la dificultad es "Fácil".
- Mantener lógica de aceptación de pedidos, pickups y dropoffs sin cambios.
- No tocar las demás dificultades.

**Juego.py**

Añadir movimiento aleatorio para dificultad "Fácil".

[Juego.py +17 -3](#)

puedes aumentar la velocidad para que sea igual a la del jugador?

[Juego.py](#)

Aumentar velocidad del NPC igual a la del jugador.

- Añadir atributos de movimiento suave del NPC (`targets` y velocidad).
- Evitar recalcular ruta mientras se mueve (early return).
- Cambiar actualización para usar movimiento suave por pixeles como el jugador.
- Sumar función `_actualizar_movimiento_npc` y llamarla en `on_update`.
- Ajustar `finally` en `actualizar_npc` para no cortar movimiento suave.

[Juego.py +55 -18](#)

GPT-5 + 1x

documenta el código añadiendo comentarios útiles para entender el código, también ya hay comentarios analizalos y fíjate si es mejor cambiar lo que está escrito o mejorar el texto.

Buenas tardes, Para mi juego Courier Quest decidí implementar niveles y un NPC que se maneje por medio de estos niveles, el nivel fácil el NPC tiene que moverse de forma Random, el nivel Medio tiene que moverse de forma Greedy, Minimax o Expectimax, el nivel difícil tiene que moverse mediante el Algoritmo de Dijkstra. todo mediante inteligencia artificial.

te dejo aqui mi codigo

si el NPC no tiene un pedido asignando entonces que esté quieto recuperando la energía, todo esto si la dificultad está en difícil

dos cosas, la resistencia del bot sigue sin bajarse bien, se queda en 99 y de ahí no baja, recuerde que debe seguir las mismas reglas que el jugador normal, segundo haga la barra de resistencia más corta para poder hacer el cuadro menos ancho

el cuadro que hizo nuevo, está muy largo, ocupo que sea un cuadro pequeño al lado izquierdo, la resistencia no se está bajando bien, todo lo demás sí se actualiza bien

ok perfecto, ahora cuando sea con bot quiero que aparezcan los stats del bot además de los del jugador, y que se le baje la resistencia igual con la misma logica que el jugador

está medio bien pq al ponerle jugar con bot me salen las reglas y el poner nombre dos veces como que se repite todo y el ponerle jugar con bot tambien, revise eso

hagalo de nuevo, algo asi como en el primer popup haya una opcion de inicializar partida con bot y otra que sea sin bot

ok now I want the bots to move as smoothless as the normal player and be affected by weather and weight as the normal player

ok el normal está bien pero revise el algoritmo del dificil pq no me está sirviendo como debe ser, es el djistra

tengo varios problemas, el bot dificil se mueve pero se va a otro lado, como que no usa el algoritmo no busca el paquete, y el bot normal se mueve un poco y despues de la nada se queda pegado

ok ahora quiero que los bots compitan por el mismo paquete, no que hayan dos paquetes y dos dropofs uno para el ot y otro para el jugador, que sea por LOS MISMO QUE SE EXTRAEN DEL API NORMAL

quiero que me arregle el npc dificil pq en este momento está haciendo como que movimientos aleatorios a partir del segundo paquete