# Explicación de la Clase Movimiento

La clase `Movimiento` gestiona el desplazamiento del personaje en el juego. Permite mover al personaje con las teclas W, A, S, D y hacer que mire en dirección al puntero del ratón. Además, gestiona la detección de colisiones contra los obstáculos del escenario y el desplazamiento de la cámara.

## 1. Definición de la Clase

Ubicación: `juegoprog.sistema.Movimiento`

Extiende: `JPanel` (se integra en la interfaz gráfica)

Implementa: `ActionListener` (para actualizar el movimiento con un `Timer`)

## 2. Explicación Detallada del Código

### Atributos

```java  
private final int SCREEN\_WIDTH = 1280;  
private final int SCREEN\_HEIGHT = 720;  
private int velocidad = 5;  
private double ang = 0;  
private boolean up, down, left, right;  
```

✔️ `SCREEN\_WIDTH` y `SCREEN\_HEIGHT`: Dimensiones de la pantalla del juego.

✔️ `velocidad`: Define la rapidez del movimiento del personaje.

✔️ `ang`: Ángulo de rotación del personaje en función del ratón.

✔️ `up, down, left, right`: Variables booleanas que indican si una tecla está presionada.

### Constructor

```java  
public Movimiento(EscenarioDistritoSombrio escenario, ColisionesPanel colisiones) { ... }  
```

✔️ Recibe como parámetros el escenario y el panel de colisiones.

✔️ Configura los `KeyListener` y `MouseMotionListener` para capturar eventos del teclado y ratón.

✔️ Inicia un `Timer` que actualiza la posición del personaje cada 16 ms.

### Método actualizarAngulo()

```java  
private void actualizarAngulo() { ... }  
```

✔️ Calcula el ángulo de rotación del personaje según la posición del ratón.

✔️ Se usa `Math.atan2()` para obtener el ángulo entre el personaje y el puntero.

### Método toggleMovement()

```java  
private void toggleMovement(int keyCode, boolean pressed) { ... }  
```

✔️ Se ejecuta cuando se presiona o suelta una tecla de movimiento (W, A, S, D).

✔️ Activa o desactiva los booleanos `up, down, left, right`.

### Método actionPerformed()

```java  
@Override  
public void actionPerformed(ActionEvent e) { ... }  
```

✔️ Se ejecuta automáticamente cada 16 ms debido al `Timer`.

✔️ Llama a `movePlayer()` para actualizar la posición del personaje.

✔️ Llama a `repaint();` para redibujar al personaje.

### Método movePlayer()

```java  
private void movePlayer() { ... }  
```

✔️ Calcula el nuevo desplazamiento del mapa en función de las teclas presionadas.

✔️ Consulta `colisiones.hayColision()` para verificar si el personaje puede moverse.

✔️ Si el borde del escenario impide el desplazamiento, mueve el personaje en su lugar.

✔️ Aplica restricciones para evitar que el personaje salga de los límites del escenario.

### Método paintComponent()

```java  
@Override  
protected void paintComponent(Graphics g) { ... }  
```

✔️ Dibuja al personaje en el centro de la pantalla.

✔️ Aplica una rotación usando `Graphics2D.rotate(ang);` para orientar el personaje hacia el ratón.

✔️ Representa al personaje como un cuadrado rojo de 20x20 píxeles.

## 3. Ejemplo de Uso

Para usar la clase `Movimiento` en el juego:

```java  
Movimiento movimiento = new Movimiento(escenario, colisiones);  
```

✔️ Se crea una instancia de `Movimiento` pasándole el escenario y las colisiones.

✔️ Se agrega al `JPanel` principal para capturar eventos de teclado y ratón.

## 4. Conclusión

La clase `Movimiento` permite mover al personaje y gestionar la cámara de forma fluida. Implementa eventos de teclado y ratón, detecta colisiones y sincroniza el desplazamiento del escenario con el movimiento del personaje.