Smart Home - Sistema de Hogar Inteligente

Clases y Responsabilidades

1. Rol (Enum)

Descripción: Enumeración que define los roles disponibles en el sistema.

Valores:

- `USUARIO = 'USUARIO'` - Rol de usuario estándar
- `ADMIN = 'ADMIN'` - Rol de administrador

---

2. Usuario

Descripción: Representa a una persona que utiliza el sistema.

Atributos:

- `id_usuario` - Identificador único del usuario (público)
- `__nombre: str` - Nombre del usuario (privado)
- `__apellido: str` - Apellido del usuario (privado)
- `__email: str` - Correo electrónico (privado)
- `__rol: Rol` - Rol asignado del Enum Rol (privado)
- `_contrasenia: str` - Contraseña del usuario (protegido)

Métodos:

- `__init__(id_usuario, nombre, apellido, email, rol: Rol, contrasenia)` - Constructor de la clase
- `__str__(): str` - Representación en string del usuario con formato
- `from_object(objeto): Usuario` - Método estático para crear un Usuario desde un objeto/tupla
- `to_object(): dict` - Convierte el usuario a un diccionario
- `from_list(lista): list[Usuario]` - Método estático para crear una lista de Usuarios desde una lista de objetos
- `is_admin(): bool` - Verifica si el usuario tiene rol de administrador

Relación: Un Usuario puede poseer múltiples Viviendas (0..*)

---

2. Vivienda

Descripción: Modela una casa o apartamento dentro del sistema.

Atributos:

- `viviendas: list[Vivienda]` - Lista de viviendas
- `contador_id: int` - Contador para IDs únicos
- `id_vivienda: int` - Identificador único de la vivienda
- `nombre: str` - Nombre de la vivienda
- `direccion: str` - Dirección física
- `ubicaciones: list[Ubicacion]` - Lista de ubicaciones/habitaciones

Métodos:

- `__init__(nombre: str, direccion: str)` - Constructor
- `get_id(): int` - Obtiene el ID
- `get_nombre(): str` - Obtiene el nombre
- `get_direccion(): str` - Obtiene la dirección
- `get_ubicaciones(): list[str]` - Obtiene la lista de ubicaciones
- `set_nombre(nuevo_nombre: str): bool` - Modifica el nombre
- `set_direccion(nueva_direccion: str): bool` - Modifica la dirección
- `agregar_ubicacion(ubicacion: Ubicacion): bool` - Agrega una ubicación
- `obtener_ubicacion(nombre: str): Ubicacion` - Obtiene una ubicación por nombre
- `eliminar_ubicacion(nombre: str): bool` - Elimina una ubicación
- `buscar_vivienda_por_nombre(nombre: str): Vivienda` - Busca vivienda por nombre
- `eliminar_vivienda(nombre: str, confirmar: str): str` - Elimina una vivienda
- `listar_viviendas(): list[dict]` - Lista todas las viviendas

Relación: Una Vivienda contiene una o más Ubicaciones (1..*) mediante composición

---

3. Ubicacion

Descripción: Zona o habitación dentro de una Vivienda (ej: "Sala de estar", "Dormitorio", "Cocina").

Atributos:

- `id_ubicacion: int` - Identificador único

- `nombre: string` - Nombre de la ubicación
- `dispositivos: list[Dispositivo]` - Lista de dispositivos en esta ubicación

Métodos:

- `_generar_siguiente_id(): int` - Genera el siguiente ID
- `__init__(nombre: str)` - Constructor
- `agregar_dispositivo(nombre_dispositivo: str): bool` - Agrega un dispositivo
- `obtener_dispositivos(): str` - Obtiene la lista de dispositivos

Relación: Una Ubicacion contiene uno o más Dispositivos (1..*) mediante composición

---

4. Dispositivo

Descripción: Representa cualquier aparato inteligente del sistema.

Atributos:

- `id_dispositivo: int` - Identificador único
- `nombre: str` - Nombre del dispositivo
- `tipo: int` - Tipo de dispositivo (enumeración)
- `estado: bool` - Estado actual (encendido/apagado)
- `TIPO_A_TEXTO: dict[int,str]` - Diccionario de conversión de tipos
- `TIPO_CAMARA: int = 1` - Constante para cámara
- `TIPO_LUZ: int = 2` - Constante para luz
- `TIPO_MUSICA: int = 3` - Constante para música

Métodos:

- `__init__(nombre: str, tipo: int, estado: bool)` - Constructor
- `nombre(): str` - Obtiene el nombre
- `tipo(): int` - Obtiene el tipo
- `estado(): bool` - Obtiene el estado
- `estado(nuevo_estado: bool): void` - Establece el estado
- `encender(): void` - Enciende el dispositivo
- `apagar(): void` - Apaga el dispositivo
- `__str__(): str` - Representación en string

Relación: Pertenece a una Ubicacion y puede ser gestionado por ControladorDispositivos y Automatizacion

---

5. ControladorDispositivos

Descripción: Gestor central para los dispositivos del sistema.

Atributos:

- `id_controlador: int` - Identificador único del controlador
- `dispositivos: list[Dispositivo]` - Lista de dispositivos gestionados

Métodos:

- `__init__() -> void` - Constructor
- `agregar_dispositivo(nombre: str, tipo: int, estado: bool): bool` - Agrega un dispositivo
- `buscar_dispositivo_por_nombre(nombre: str): void` - Busca un dispositivo
- `eliminar_dispositivo(nombre: str): bool` - Elimina un dispositivo
- `listar_dispositivos(): void` - Lista todos los dispositivos

Relación: Gestiona múltiples Dispositivos (0..*)

---

6. Automatizacion

Descripción: Representa rutinas o reglas para controlar varios dispositivos de forma automática.

Atributos:

- `id_automatizacion: int` - Identificador único
- `dispositivos: list[Dispositivo]` - Lista de dispositivos involucrados
- `hora_activacion_modo_noche: int` - Hora de activación del modo noche

Métodos:

- `__init__(dispositivos: list)` - Constructor
- `get_dispositivos(): list` - Obtiene los dispositivos
- `get_hora_activacion_modo_noche(): int` - Obtiene la hora configurada
- `set_dispositivos(dispositivos: list): bool` - Establece los dispositivos
- `set_hora_activacion_modo_noche(nueva_hora: int): bool` - Configura la hora
- `consultar_automatizaciones(self): tuple` - Consulta configuraciones
- `mostrar_estado: str` - Muestra el estado actual
- `activar_modo_fiesta(): str` - Activa modo fiesta
- `apagar_modo_fiesta(): str` - Apaga modo fiesta

- `activar_modo_noche(): str` - Activa modo noche
- `apagar_modo_noche(): str` - Apaga modo noche
- `configurar_hora_modo_noche(nueva_hora: int): bool` - Configura hora
- `verificar_hora_modo_noche(): int` - Verifica la hora configurada

Relación: Se aplica sobre uno o más Dispositivos (1..*)

Relaciones Clave del Sistema

1. **Usuario → Vivienda:** Un Usuario puede poseer múltiples Viviendas (0..*)

2. **Vivienda → Ubicacion:** Una Vivienda contiene una o más Ubicaciones (1..*) mediante **composición** (las ubicaciones no existen sin la vivienda)

3. **Ubicacion → Dispositivo:** Una Ubicacion contiene uno o más Dispositivos (1..*) mediante **composición**

4. **ControladorDispositivos → Dispositivo:** Gestiona múltiples dispositivos del sistema (relación 1 a 0..*)

5. **Automatizacion → Dispositivo:** Una automatización controla uno o más dispositivos (relación 1 a 0..*)

Tipos de Dispositivos (Enumeración)

- **TIPO_CAMARA = 1:** Dispositivos de vigilancia
- **TIPO_LUZ = 2:** Sistemas de iluminación
- **TIPO_MUSICA = 3:** Sistemas de audio

Roles de Usuario (Enumeración)

Aunque no aparece como clase separada en el diagrama, el sistema maneja roles mediante un Enum:

- **USUARIO = 1:** Permisos básicos
- **ADMIN = 2:** Permisos administrativos (puede modificar roles)
- **DUEÑO = 3:** Propietario de viviendas

Notas de Implementación

- El sistema utiliza composición para las relaciones Vivienda-Ubicacion y Ubicacion-Dispositivo, lo que significa que las ubicaciones y dispositivos no pueden existir independientemente de sus contenedores.

- Los identificadores (IDs) se generan automáticamente mediante contadores internos en cada clase.

- La clase Automatizacion implementa modos predefinidos (modo fiesta, modo noche) que coordinan múltiples dispositivos simultáneamente.

- El ControladorDispositivos actúa como un gestor centralizado, permitiendo operaciones CRUD sobre los dispositivos del sistema.