Project Description: Elevator Simulation

This project is an elevator simulation game implemented using Pygame. The simulation includes multiple floors and elevators, each responding to user inputs to simulate the operation of elevators in a building. The core components of the project are the `Building`, `Elevator`, `Floor`, and `Timer` classes, each responsible for specific functionalities within the simulation.

Classes and Their Responsibilities

1. **Building Class**:

   When a `Building` instance is created, it separately instantiates the elevators and floors.

   - **Responsibilities**:
     - Manages the overall structure and behavior of the building, including the floors and elevators.
     - Handles user inputs to call elevators and updates the state of the building.
     - Draws the building, floors, and elevators on the Pygame window.
   - **Key Methods**:
     - `identifies_clicks(click)`: Identifies which floor button was clicked and calls an elevator to that floor.
     - `call_elevator(floor)`: Determines which elevator should respond to a floor call. The function applies the `get_call` method from the chosen elevator and updates the floor with the estimated waiting time.
     - `draw_and_update_all(window)`: Draws and updates the state of all floors and elevators in the building.
2. **Elevator Class**:
   - **Responsibilities**:
     - Simulates the behavior of an individual elevator, including movement between floors, responding to calls, and playing sound on arrival.
     - Manages its own position, direction of motion, and stops.
   - **Key Methods**:
     - `time_to_floor(floor)`: Calculates the time required to reach a specified floor.
     - `get_call(floor)`: Adds a floor call to the elevator's queue.
     - `update_position()`: Updates the elevator's position based on time and movement direction.
     - `draw_elevator(window)`: Draws the elevator on the Pygame window.
3. **Floor Class**:
   - **Responsibilities**:
     - Represents an individual floor in the building.

        - Manages the state of its call button and displays a timer indicating how long until an elevator arrives.
    - **Key Methods**:
        - `color_button()`: Determines the color of the floor button based on whether an elevator is on its way.
        - `draw_floor(window)`: Draws the floor, its button, and the timer on the Pygame window.
4. **Timer Class**:
    - **Responsibilities**:
        - Manages countdown timers for various events in the simulation, such as elevator movements and delays.
    - **Key Methods**:
        - `time_left()`: Calculates and returns the remaining time on the timer.

Main Script

The main script initializes the Pygame environment and creates an instance of the `Building` class. It handles the main game loop, which includes:

- Capturing user input events.
- Updating the state of the building and its components.
- Rendering the building, floors, and elevators on the screen.

The game runs in a loop until the user decides to quit, ensuring that the simulation continues to respond to inputs and update in real-time.