Tile Merging

Tile merging in the "2248" game follows specific rules to combine adjacent tiles with the same value into a single tile:

1. **Selection of Tiles:**
    - When a player clicks on a tile (or drags over multiple tiles), they are selected for potential merging.
2. **Merge Process:**
    - When the player releases the mouse after selecting tiles, the game checks if the selected tiles can be merged.
    - It calculates the sum of the values of all selected tiles.
3. **Nearest Power of 2:**
    - The game then finds the nearest power of 2 less than or equal to this sum using math.floor(math.log2(sum_value)).
    - If the sum itself is a power of 2, that exact value is retained.
4. **Resulting Tile:**
    - The resulting merged tile is placed in a randomly selected empty spot on the board.
    - If there are no empty spots left, the game checks for potential moves that would allow further merging.
5. **Score Update:**
    - After merging, the player's score is updated by adding the sum of the merged tiles.

Score Calculation

The score in the "2248" game is calculated based on the values of tiles that are merged:

- **Initialization:** The score starts at zero when the game begins or is reset.
- **Merging Tiles:** Each time tiles are merged, the sum of their values contributes to the score.
- **Display Update:** After each merge, the GUI updates to display the current score using self.update_score().

Game-Over Conditions

The game can end under two main conditions:

1. **No More Moves:**
    - If there are no empty spots left on the board and no adjacent tiles have the same value that can be merged, the game is over.
    - This condition is checked in the check_game_over() method.
2. **Winning Condition:**
    - If a tile with the value 2048 (or any other target value set) appears on the board, the player wins.
    - This condition is checked in the check_win() method.

Implementation Details

- **Tile Merging:** The merge_selected_tiles() method handles the merging logic by summing selected tiles, finding the nearest power of 2, updating the board, adding new tiles, updating the score, and checking win/game-over conditions.
- **Score Update:** The update_score() method updates the displayed score label whenever the score changes due to tile merges.
- **Game-Over Check:** The check_game_over() method iterates through the board to check for adjacent tiles with the same value or for any remaining empty tiles. If no moves are possible, it returns True, indicating the game is over.

Code explaination:

Imports:

```
import tkinter as tk
import random
import math
```

- tkinter: Used for creating the GUI (Graphical User Interface).
- random: Used for generating random numbers.
- math: Used for mathematical operations.

Class Game2248:

- This class manages the entire game logic and GUI interactions.
- **Initialization (__init__ method):**
  - Sets up the game window (master), title, and initial game settings.
  - Defines frames (start_frame, game_frame, gameover_frame) for different game states (start screen, game screen, game over screen).
  - Creates various GUI widgets like labels, buttons, and canvas for drawing the game board.

Game Methods:

- start_game(): Switches the frame to the game screen and resets the game.
- reset_game(): Initializes or resets the game state, including the game board, score, and initial tiles.
- prefill_board(): Fills the game board with initial tiles (values like 2, 4, 8, etc.).
- draw_board(): Draws the game board on the canvas, including tiles with their respective values and colors.
- get_tile_color(): Returns a color based on the tile value.
- on_click(), on_drag(), on_release(): Methods for handling mouse interactions (selecting, dragging, releasing tiles).
- highlight_selected_tiles(): Highlights selected tiles.
- merge_selected_tiles(): Merges selected tiles based on game rules, updates the board, score, and checks for win or game over conditions.
- add_random_tiles(): Adds random tiles after merging.
- check_win(), check_game_over(): Checks if the player has won or if the game is over.
- update_score(): Updates the displayed score.

Helper Methods:

- switch_frame(): Switches between different frames (start, game, game over).

Main Execution:

```
python
Copy code
if __name__ == "__main__":
    root = tk.Tk()
    game = Game2248(root)
    root.mainloop()
```

- Creates an instance of tkinter.Tk() as the main application window.
- Creates an instance of Game2248, passing the main window as an argument.
- Starts the tkinter event loop (root.mainloop()), which handles user inputs and updates the GUI.