DEV-Dokumentáció
Tánczos János - SNAKE
GWVABC

File tree:
Main.c
Lib/
- Game/
  - game.c - game.h
  - snake.c - snake.h
- Menu/
  - Menu.c - menu.h
  - button.c - button.h
- rendering.c - rendering.h

game.c -Game.h:

```c
/// Full game loop, handles everything from rendering and game logic
/// @param renderer [in] constraints the SDL_renderer
enum WindowState StartGame(GameRenderer *renderer);
```

snake.c - snake.h

```c
/// Direction
enum Direction {
   UP,
   DOWN,
   LEFT,
   RIGHT
};
/// A linked list as the snake
typedef struct Snake {
   Vector2 bodyPart;
   struct Snake *next;
} Snake;

/// Creates the snake from scratch
/// @param startPos the coordinates of the whole Snake
/// @param length the initialization length if the Snake
/// @returns A Snake struct what it self is a linked list
/// @attention You have to Free the snake with the FreeSnake function
Snake *CreateSnake(Vector2 startPos, int length);

/// Moves the snake to the given direction
/// @param snake the snake is Self
/// @param nextDirection direction of the move
bool MoveSnake(Snake *snake, enum Direction next);

/// Returns the last body part's postion
/// @param snake
/// @returns true if the move can be done, and false if something is blocking the snake it
self
Vector2 LastSnakeBody(Snake *snake);

/// Frees the snake
/// @param snake
void FreeSnake(Snake *snake);

/// Expands the snake to the given direction
/// @param snake the snake is Self
/// @param nextDirection direction of the expansion
void ExpandSnake(Snake *snake, enum Direction nextDirection);
```

rendering.c - rendering.h

```c
/// States of the app window
enum WindowState {
   GAME,
   MENU,
   SCORE_BOARD,
   EXIt
};
/// Constraints everything what essential info tu rendering
typedef struct GameRenderer {
   SDL_Renderer *renderer;
   enum WindowState state;
} GameRenderer;

/// A 2D vector with X and Y coordinates
typedef struct Vector2 {
   int x, y;
} Vector2;

/// Initialize the renderer this step makes the program graphical
/// @return a GameRenderer object what used in every other rendering specific task
GameRenderer InitGameRenderer();
```

Menu.c - menu.h

```c
/// States of the app window
enum WindowState {
   GAME,
   MENU,
   SCORE_BOARD,
   EXIt
};
/// Constraints everything what essential info tu rendering
typedef struct GameRenderer {
   SDL_Renderer *renderer;
   enum WindowState state;
} GameRenderer;

/// A 2D vector with X and Y coordinates
typedef struct Vector2 {
   int x, y;
} Vector2;

/// Initialize the renderer this step makes the program graphical
/// @return a GameRenderer object what used in every other rendering specific task
GameRenderer InitGameRenderer();
```