Carcassone Clone

Készítette Doxygen 1.9.4

Chapter 1

Adatszerkezet-mutató

1.1 Adatszerkezetek

Az összes adatszerkezet listája rövid leírásokkal:

Chapter 2

Fájlmutató

2.1 Fájllista

Az összes fájl listája rövid leírásokkal:

Chapter 3

Adatszerkezetek dokumentációja

3.1 Button struktúrareferencia

`#include <ui.h>`

Adatmezők

- char * label
- SDL_Texture * label_texture
- SDL_Rect label_rect
- TTF_Font * used_font
- SDL_Rect local_rect
- SDL_Rect global_rect
- SDL_Color bg_color

3.1.1 Részletes leírás

Gomb

3.1.2 Adatmezők dokumentációja

3.1.2.1 bg_color

`SDL_Color Button::bg_color`

3.1.2.2 global_rect

```
SDL_Rect Button::global_rect
```

3.1.2.3 label

```
char* Button::label
```

3.1.2.4 label_rect

```
SDL_Rect Button::label_rect
```

3.1.2.5 label_texture

```
SDL_Texture* Button::label_texture
```

3.1.2.6 local_rect

```
SDL_Rect Button::local_rect
```

3.1.2.7 used_font

```
TTF_Font* Button::used_font
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/ui.h

3.2 Carcassone struktúrareferencia

```
#include <app.h>
```

Adatmezők

- int width
- int height
- bool is_running
- SDL_Window * window
- SDL_Surface * window_icon
- SDL_Texture * splash_title
- SDL_Renderer * renderer
- TTF_Font * default_font
- TTF_Font * small_font
- AppState state
- MenuScreen * menu_screen
- LeaderboardScreen * lboard_screen
- GameScreen * game_screen
- SDL_mutex * smutex

3.2.1 Részletes leírás

Magát a programot reprezentáló struktúra.

3.2.2 Adatmezők dokumentációja

3.2.2.1 default_font

```
TTF_Font* Carcassone::default_font
```

3.2.2.2 game_screen

```
GameScreen* Carcassone::game_screen
```

3.2.2.3 height

```
int Carcassone::height
```

3.2.2.4 is_running

```
bool Carcassone::is_running
```

3.2.2.5 lboard_screen

```
LeaderboardScreen* Carcassone::lboard_screen
```

3.2.2.6 menu_screen

```
MenuScreen* Carcassone::menu_screen
```

3.2.2.7 renderer

```
SDL_Renderer* Carcassone::renderer
```

3.2.2.8 small_font

```
TTF_Font * Carcassone::small_font
```

3.2.2.9 smutex

```
SDL_mutex* Carcassone::smutex
```

3.2.2.10 splash_title

```
SDL_Texture* Carcassone::splash_title
```

3.2.2.11 state

```
AppState Carcassone::state
```

3.2.2.12 width

```
int Carcassone::width
```

3.2.2.13 window

```
SDL_Window* Carcassone::window
```

3.2.2.14 window_icon

```
SDL_Surface* Carcassone::window_icon
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/app.h

3.3 CardPile struktúrareferencia

```
#include <tile.h>
```

Adatmezők

- TileType card
- struct CardPile * next

3.3.1 Részletes leírás

Kártyapakli stackként implementálva.

3.3.2 Adatmezők dokumentációja

3.3.2.1 card

```
TileType CardPile::card
```

3.3.2.2 next

```
struct CardPile* CardPile::next
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/tile.h

3.4 GameScreen struktúrareferencia

`#include <ui.h>`

Adatmezők

- bool is_ready
- bool is_game_over
- int held_arrow_keys [4]
- SDL_FPoint board_offset
- Tile ** board
- SDL_Texture * board_texture
- Tile * drawn_tile
- CardPile * card_pile
- size_t pile_index
- SDL_Texture * pile_counter
- bool update_counter
- Player players [2]
- Player * curr_player
- Player * winner
- SDL_Texture * crown_texture
- SDL_Texture * player_input_labels [2]
- Prompt player_name_inputs [2]
- Prompt * active_input
- Button ready_button
- Button end_turn_button
- Button concede_button
- TilesetWrapper tileset_wrapper

3.4.1 Részletes leírás

Játéknézet.

3.4.2 Adatmezők dokumentációja

3.4.2.1 active_input

```
Prompt* GameScreen::active_input
```

3.4.2.2 board

```
Tile** GameScreen::board
```

3.4.2.3 board_offset

```
SDL_FPoint GameScreen::board_offset
```

3.4.2.4 board_texture

```
SDL_Texture* GameScreen::board_texture
```

3.4.2.5 card_pile

```
CardPile* GameScreen::card_pile
```

3.4.2.6 concede_button

```
Button GameScreen::concede_button
```

3.4.2.7 crown_texture

```
SDL_Texture* GameScreen::crown_texture
```

3.4.2.8 curr_player

```
Player* GameScreen::curr_player
```

3.4.2.9 drawn_tile

```
Tile* GameScreen::drawn_tile
```

3.4.2.10 end_turn_button

```
Button GameScreen::end_turn_button
```

3.4.2.11 held_arrow_keys

```
int GameScreen::held_arrow_keys[4]
```

3.4.2.12 is_game_over

```
bool GameScreen::is_game_over
```

3.4.2.13 is_ready

```
bool GameScreen::is_ready
```

3.4.2.14 pile_counter

```
SDL_Texture* GameScreen::pile_counter
```

3.4.2.15 pile_index

```
size_t GameScreen::pile_index
```

3.4.2.16 player_input_labels

```
SDL_Texture* GameScreen::player_input_labels[2]
```

3.4.2.17 player_name_inputs

```
Prompt GameScreen::player_name_inputs[2]
```

3.4.2.18 players

```
Player GameScreen::players[2]
```

3.4.2.19 ready_button

```
Button GameScreen::ready_button
```

3.4.2.20 tileset_wrapper

```
TilesetWrapper GameScreen::tileset_wrapper
```

3.4.2.21 update_counter

```
bool GameScreen::update_counter
```

3.4.2.22 winner

```
Player* GameScreen::winner
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/ui.h

3.5 Leaderboard struktúrareferencia

```
#include <player.h>
```

Adatmezők

- char const * records_file_path
- LeaderboardEntry * entries
- size_t entries_size

3.5.1 Részletes leírás

Dicsőséglista.

3.5.2 Adatmezők dokumentációja

3.5.2.1 entries

```
LeaderboardEntry* Leaderboard::entries
```

3.5.2.2 entries_size

```
size_t Leaderboard::entries_size
```

3.5.2.3 records_file_path

```
char const* Leaderboard::records_file_path
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/player.h

3.6 LeaderboardEntry struktúrareferencia

```
#include <player.h>
```

Adatmezők

- char name [(MAX_PLAYER_NAME_LEN+1) ∗sizeof(wchar_t)]
- unsigned int highscore

3.6.1 Részletes leírás

Dicsőséglista rekord.

3.6.2 Adatmezők dokumentációja

3.6.2.1 highscore

```
unsigned int LeaderboardEntry::highscore
```

3.6.2.2 name

```
char LeaderboardEntry::name[(MAX_PLAYER_NAME_LEN+1) *sizeof(wchar_t)]
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/player.h

3.7 LeaderboardScreen struktúrareferencia

```
#include <ui.h>
```

Adatmezők

- Leaderboard * leaderboard
- char syntax_error_msg [128+1]
- SDL_Texture * list_texture
- Button back_button

3.7.1 Részletes leírás

Dicsőséglistanézet.

3.7.2 Adatmezők dokumentációja

3.7.2.1 back_button

```
Button LeaderboardScreen::back_button
```

3.7.2.2 leaderboard

```
Leaderboard* LeaderboardScreen::leaderboard
```

3.7.2.3 list_texture

```
SDL_Texture* LeaderboardScreen::list_texture
```

3.7.2.4 syntax_error_msg

```
char LeaderboardScreen::syntax_error_msg[128+1]
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/ui.h

3.8 Meeple struktúrareferencia

```
#include <meeple.h>
```

Adatmezők

- bool is_placed
- int x
- int y
- SDL_Texture * texture

3.8.1 Részletes leírás

Alattvaló.

3.8.2 Adatmezők dokumentációja

3.8.2.1 is_placed

```
bool Meeple::is_placed
```

3.8.2.2 texture

```
SDL_Texture* Meeple::texture
```

3.8.2.3 x

```
int Meeple::x
```

3.8.2.4 y

```
int Meeple::y
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/meeple.h

3.9 MenuScreen struktúrareferencia

```
#include <ui.h>
```

Adatmezők

- SDL_Texture * background
- SDL_Rect button_container
- Button start_button
- Button lboard_button

3.9.1 Részletes leírás

Menünézet.

3.9.2 Adatmezők dokumentációja

3.9.2.1 background

```
SDL_Texture* MenuScreen::background
```

3.9.2.2 button_container

```
SDL_Rect MenuScreen::button_container
```

3.9.2.3 lboard_button

```
Button MenuScreen::lboard_button
```

3.9.2.4 start_button

```
Button MenuScreen::start_button
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/ui.h

3.10 Player struktúrareferencia

```
#include <player.h>
```

Adatmezők

- char name [(MAX_PLAYER_NAME_LEN+1) ∗sizeof(wchar_t)]
- unsigned int score
- bool has_placed_card
- Meeple meeples [MAX_MEEPLES]
- size_t meeples_at_hand
- SDL_Texture ∗ score_counter
- SDL_Texture ∗ handle_texture
- SDL_Texture ∗ stat_panel
- SDL_Texture ∗ own_meeple_texture
- bool update_score

3.10.1 Adatmezők dokumentációja

3.10.1.1 handle_texture

```
SDL_Texture* Player::handle_texture
```

3.10.1.2 has_placed_card

```
bool Player::has_placed_card
```

3.10.1.3 meeples

```
Meeple Player::meeples[MAX_MEEPLES]
```

3.10.1.4 meeples_at_hand

```
size_t Player::meeples_at_hand
```

3.10.1.5 name

```
char Player::name[(MAX_PLAYER_NAME_LEN+1) *sizeof(wchar_t)]
```

3.10.1.6 own_meeple_texture

```
SDL_Texture* Player::own_meeple_texture
```

3.10.1.7 score

```
unsigned int Player::score
```

3.10.1.8 score_counter

```
SDL_Texture* Player::score_counter
```

3.10.1.9 stat_panel

```
SDL_Texture* Player::stat_panel
```

3.10.1.10 update_score

```
bool Player::update_score
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/player.h

3.11 Prompt struktúrareferencia

`#include <ui.h>`

Adatmezők

- Button prompt

3.11.1 Részletes leírás

Szöveginput. (egy régi verzió miatt van külön a gombtól)

3.11.2 Adatmezők dokumentációja

3.11.2.1 prompt

`Button Prompt::prompt`

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/ui.h

3.12 Tile struktúrareferencia

`#include <tile.h>`

Adatmezők

- TileType type
- ConnectionType connections [4]
- SDL_FPoint local_coords
- SDL_FPoint global_coords
- SDL_Point board_coords
- unsigned short rotation
- bool rotatable
- bool is_scored
- bool is_expired

3.12.1 Részletes leírás

Mezőkártya.

3.12.2 Adatmezők dokumentációja

3.12.2.1 board_coords

```
SDL_Point Tile::board_coords
```

3.12.2.2 connections

```
ConnectionType Tile::connections[4]
```

3.12.2.3 global_coords

```
SDL_FPoint Tile::global_coords
```

3.12.2.4 is_expired

```
bool Tile::is_expired
```

3.12.2.5 is_scored

```
bool Tile::is_scored
```

3.12.2.6 local_coords

```
SDL_FPoint Tile::local_coords
```

3.12.2.7 rotatable

```
bool Tile::rotatable
```

3.12.2.8 rotation

```
unsigned short Tile::rotation
```

3.12.2.9 type

```
TileType Tile::type
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/tile.h

3.13 TilesetWrapper struktúrareferencia

```
#include <tile.h>
```

Adatmezők

- SDL_Texture * tile_set

3.13.1 Részletes leírás

Egy segédstruktúra a textúra atlasz tárolásához.

3.13.2 Adatmezők dokumentációja

3.13.2.1 tile_set

```
SDL_Texture* TilesetWrapper::tile_set
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/game/tile.h

Chapter 4

Fájlok dokumentációja

4.1 include/app.h fájlreferencia

```
#include <bits/pthreadtypes.h>
#include <stdbool.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "game/tile.h"
#include "game/player.h"
#include "ui.h"
```

Adatszerkezetek

- struct Carcassone

Makródefiníciók

- #define FPS 60
- #define MIN(x, y) ((x < y) ? x : y)
- #define COLOR_BLUE 25, 30, 91, 255
- #define COLOR_LIGHTBLUE 153, 204, 255, 255
- #define COLOR_WHITE 255, 255, 255, 255
- #define COLOR_RED 255, 0, 0, 255
- #define COLOR_SALMON 255, 145, 164, 255
- #define COLOR_BG 102, 102, 153, 255
- #define DBG_LOG(x, ...) SDL_LogDebug(SDL_LOG_CATEGORY_CUSTOM, x, ##__VA_ARGS__)

Enumerációk

- enum AppState { MENU , GAME , LEADERBOARD }

Függvények

- Carcassone * Carcassone__construct (int, int, char const *)

  Initicializálja az összes nézetet, SDL és TTF kontextusokat.
- void Carcassone__destroy (Carcassone *)

  Felszabadítja a megadott `Carcassone` struktúra által lefoglalt memóriát.
- void Carcassone__switch_state (Carcassone *, AppState)

  Megváltoztatja a program nézetét.
- void Carcassone__run (Carcassone *)

  A fő programciklus.
- void Carcassone__render_splash_title (Carcassone *, SDL_Rect *)

  A splash cím renderelése.
- void Carcassone__Menu__construct (Carcassone *)

  Létrehozza a menünézetet.
- void Carcassone__Menu__destroy (Carcassone *)

  Felszabadítja a megadott `Carcassone` struktúrához tartozó `MenuScreen` által lefoglalt memóriát.
- void Carcassone__Menu__render (Carcassone *)

  Menünézet megjelenítése.
- void Carcassone__Menu__handle_input (Carcassone *)

  Inputok kezelése menünézetben.
- void Carcassone__Lboard__construct (Carcassone *)

  Létrehozza a dicsőséglistanézetet.
- void Carcassone__Lboard__destroy (Carcassone *)

  Felszabadítja a megadott `Carcassone` struktúrához tartozó `LeaderboardScreen` által lefoglalt memóriát.
- void Carcassone__Lboard__render (Carcassone *)

  Dicsőséglistanézet megjelenítése.
- void Carcassone__Lboard__init_list_texture (Carcassone *)

  Létrehozza a dicsőséglistanézethez a rekordokat.
- void Carcassone__Lboard__handle_input (Carcassone *)

  Inputok kezelése dicsőglistanézetben.
- void Carcassone__Game__construct (Carcassone *)

  Létrehozza a játéknézetet.
- void Carcassone__Game__destroy (Carcassone *)

  Felszabadítja a megadott `Carcassone` struktúrához tartozó `GameScreen` által lefoglalt memóriát.
- void Carcassone__Game__handle_input (Carcassone *, float)

  Inputok kezelése játéknézetben.
- void Carcassone__Game__init_players (Carcassone *)

  Játékosok létrehozása.
- void Carcassone__Game__init_pile (Carcassone *)

  Kártyapakli létrehozása.
- void Carcassone__Game__init_board (Carcassone *)

  Játéktábla létrehozása.
- void Carcassone__Game__init_counter (Carcassone *)
- void Carcassone__Game__render (Carcassone *)

  Játéknézet megjelenítése.
- void Carcassone__Game__render_board (Carcassone *)

  Játéktábla renderelése.
- void Carcassone__Game__render_drawn_tile (Carcassone *)

  A pakli tetején levő kártya és a pakli méretének renderelése.
- void Carcassone__Game__render_player_stats (Carcassone *)

  A splash cím renderelése.

- void Carcassone__Game__render_meeples (Carcassone *)

  A lehelyezett alattvalók renderelése.
- void Carcassone__Game__render_game_over (Carcassone *)

  A "game over" állapot megjelenítése.
- void Carcassone__Game__wrapup (Carcassone *, bool)

  "Game over" szituáció menedzselése.
- void Carcassone__Game__move_board (Carcassone *, float)

  Játéktábla mozgatása.
- bool Carcassone__Game__draw_new (Carcassone *)

  Új kártya húzása a pakli tetejéről.
- bool Carcassone__Game__check_names_valid (Carcassone *)

  Leellenőrzi, hogy a két megadott játékos név helyes e.
- void Carcassone__Game__indicate_possible_placements (Carcassone *)
- bool Carcassone__Game__check_surrounding_tiles (Carcassone *, SDL_Point)

  Helyes pozíció ellenőrzése.
- void Carcassone__Game__calculate_points (Carcassone *)

  A megfelelő pontszámító függvényeknek delegálja a feladatot.
- bool Carcassone__Game__check_if_possible (Carcassone *)

  Megnézi, hogy a húzott kártyát valahova le lehet e tenni.
- void Carcassone__Game__calculate_scores_for_cloister (Carcassone *, Tile *)

  Kolostorok pontszámítása.
- void Carcassone__Game__calculate_scores_for_graph (Carcassone *, Tile *, ConnectionType)

  Utak és várak pontszámítása.
- Button Carcassone__Button__construct (Carcassone *, TTF_Font *, char *, SDL_Rect, SDL_Color, SDL↩ _Color, bool)

  Létrehoz egy gombot.
- void Carcassone__Button__destroy (Carcassone *, Button *)

  Felszabadítja a megadott `Button` struktúra által lefoglalt memóriát.
- void Carcassone__Button__render (Carcassone *, Button *)

  A gomb megjelenítése.
- bool Carcassone__Button__hover (Carcassone *, Button *, SDL_Point)
- Prompt Carcassone__Prompt__construct (Carcassone *, TTF_Font *, char *, SDL_Rect, SDL_Color, SDL↩ _Color)

  Létrehoz egy szöveginputot.
- void Carcassone__Prompt__destroy (Carcassone *, Prompt *)

  Felszabadítja a megadott `Prompt` struktúra által lefoglalt memóriát.
- void Carcassone__Prompt__render (Carcassone *, Prompt *)

  A szöveginput megjelenítése.
- void Carcassone__Prompt__edit (Carcassone *, Prompt *, char *, bool)

  Vagy hozzátold a szöveginput címkéjéhez, vagy lecseréli azt.

4.1.1 Makródefiníciók dokumentációja

4.1.1.1 COLOR_BG

```
#define COLOR_BG 102, 102, 153, 255
```

4.1.1.2 COLOR_BLUE

```
#define COLOR_BLUE 25, 30, 91, 255
```

Színek.

4.1.1.3 COLOR_LIGHTBLUE

```
#define COLOR_LIGHTBLUE 153, 204, 255, 255
```

4.1.1.4 COLOR_RED

```
#define COLOR_RED 255, 0, 0, 255
```

4.1.1.5 COLOR_SALMON

```
#define COLOR_SALMON 255, 145, 164, 255
```

4.1.1.6 COLOR_WHITE

```
#define COLOR_WHITE 255, 255, 255, 255
```

4.1.1.7 DBG_LOG

```
#define DBG_LOG(
              x,
              ...  ) SDL_LogDebug(SDL_LOG_CATEGORY_CUSTOM, x, ##__VA_ARGS__)
```

Development alatt használt logger.

4.1.1.8 FPS

```
#define FPS 60
```

Cél FPS.

4.1.1.9 MIN

```
#define MIN(
              x,
              y ) ((x < y) ?  x :  y)
```

Egyszerű minimum meghatározó.

4.1.2 Enumerációk dokumentációja

4.1.2.1 AppState

```
enum AppState
```

A program állapotai: menü, játék, dicsőséglista.

Enumeráció-értékek

MENU	
GAME	
LEADERBOARD	

4.1.3 Függvények dokumentációja

4.1.3.1 Carcassone__Button__construct()

```
Button Carcassone__Button__construct (
            Carcassone * this,
            TTF_Font * font,
            char * label,
            SDL_Rect global_rect,
            SDL_Color bg_color,
            SDL_Color text_color,
            bool center )
```

Létrehoz egy gombot.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Button__destroy` függvényt.

Paraméterek

this	A renderert tartalmazó `Carcassone` struktúra.
font	A címkéhez használt betűtípus.

Paraméterek

label	A címke tartalma.
global_rect	A gomb globális pozíciója (az ablakon).
bg_color	A gomb színe.
text_color	A címke színe.
center	A container-ben középre legyen e igazítva a címke.

Visszatérési érték

Az új `Button`.

4.1.3.2 Carcassone__Button__destroy()

```
void Carcassone__Button__destroy (
            Carcassone * this,
            Button * button )
```

Felszabadítja a megadott `Button` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek az ablakára az adott gomb renderelve van (nem használt, régi verzió miatt).
button	A `Button` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.3 Carcassone__Button__hover()

```
bool Carcassone__Button__hover (
            Carcassone * ,
            Button * ,
            SDL_Point  )
```

4.1.3.4 Carcassone__Button__render()

```
void Carcassone__Button__render (
            Carcassone * this,
            Button * button )
```

A gomb megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amelyhez a renderer tartozik.
button	A megjelenítendő gomb.

4.1.3.5 Carcassone__construct()

```
Carcassone * Carcassone__construct (
             int width,
             int height,
             char const * title )
```

Initicializálja az összes nézetet, SDL és TTF kontextusokat.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__destroy` függvényt.

Paraméterek

width	Az ablak szélessége.
height	Az ablak magassága.
title	Az ablak címe.

Visszatérési érték

Pointer az újonnan létrehozott `Carcassone` structra.

4.1.3.6 Carcassone__destroy()

```
void Carcassone__destroy (
             Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.7 Carcassone__Game__calculate_points()

```
void Carcassone__Game__calculate_points (
             Carcassone * this )
```

A megfelelő pontszámító függvényeknek delegálja a feladatot.

Paraméterek

this	

4.1.3.8 Carcassone__Game__calculate_scores_for_cloister()

```
void Carcassone__Game__calculate_scores_for_cloister (
            Carcassone * this,
            Tile * tile )
```

Kolostorok pontszámítása.

Paraméterek

this	
tile	

4.1.3.9 Carcassone__Game__calculate_scores_for_graph()

```
void Carcassone__Game__calculate_scores_for_graph (
            Carcassone * this,
            Tile * tile,
            ConnectionType conn_type )
```

Utak és várak pontszámítása.

Paraméterek

this	
tile	
conn_type	

4.1.3.10 Carcassone__Game__check_if_possible()

```
bool Carcassone__Game__check_if_possible (
            Carcassone * this )
```

Megnézi, hogy a húzott kártyát valahova le lehet e tenni.

Paraméterek

this	

Visszatérési érték

Le lehet e tenni a húzott kártyát valahova.

4.1.3.11 Carcassone__Game__check_names_valid()

```
bool Carcassone__Game__check_names_valid (
            Carcassone * this )
```

Leellenőrzi, hogy a két megadott játékos név helyes e.

Ellenőrzi a hosszúságukat és hogy nem e ugyanazok.

Paraméterek

this	A `Carcassone` struktúra, amihez a nézet tartozik.

Visszatérési érték

Helyesek e.

4.1.3.12 Carcassone__Game__check_surrounding_tiles()

```
bool Carcassone__Game__check_surrounding_tiles (
            Carcassone * this,
            SDL_Point tcoords )
```

Helyes pozíció ellenőrzése.

Paraméterek

this	A `Carcassone struktúra`, ami tartalmazza a játéknézetet.
tcoords	A kapott kártya potenciális helye a táblán.

Visszatérési érték

Letehető-e a megfelelő pozícióba az adott kártya.

4.1.3.13 Carcassone__Game__construct()

```
void Carcassone__Game__construct (
            Carcassone * this )
```

Létrehozza a játéknézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a Carcassone__Game__↩ destroy függvényt.

Paraméterek

this	A Carcassone struktúra, amelynek létrehozza a játéknézetét.

4.1.3.14 Carcassone__Game__destroy()

```
void Carcassone__Game__destroy (
            Carcassone * this )
```

Felszabadítja a megadott Carcassone struktúrához tartozó GameScreen által lefoglalt memóriát.

Paraméterek

this	A Carcassone struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.15 Carcassone__Game__draw_new()

```
bool Carcassone__Game__draw_new (
            Carcassone * this )
```

Új kártya húzása a pakli tetejéről.

Paraméterek

this	A Carcassone struktúra.

Visszatérési érték

Lehet e még húzni (ha üres a pakli: false).

4.1.3.16 Carcassone__Game__handle_input()

```
void Carcassone__Game__handle_input (
            Carcassone * this,
            float dt )
```

Inputok kezelése játéknézetben.

Paraméterek

this	A `Carcassone` struktúra, amihez tartozik a játéknézet.
dt	Deltaidő.

4.1.3.17 Carcassone__Game__indicate_possible_placements()

```
void Carcassone__Game__indicate_possible_placements (
            Carcassone * this )
```

4.1.3.18 Carcassone__Game__init_board()

```
void Carcassone__Game__init_board (
            Carcassone * this )
```

Játéktábla létrehozása.

Létrehozza a játéktáblát és le is helyezi a kezdőkártyát.

Paraméterek

this	A `Carcassone` struktúra.

4.1.3.19 Carcassone__Game__init_counter()

```
void Carcassone__Game__init_counter (
            Carcassone *  )
```

4.1.3.20 Carcassone__Game__init_pile()

```
void Carcassone__Game__init_pile (
            Carcassone * this )
```

Kártyapakli létrehozása.

Véletlenszerűen megkeveri a paklit és létrehozza a mezőkártyákat mindegyikhez.

Paraméterek

this	A `Carcassone` struktúra.

4.1.3.21 Carcassone__Game__init_players()

```
void Carcassone__Game__init_players (
            Carcassone * this )
```

Játékosok létrehozása.

Inicializálja a két játékost előre megadott adatok alapján (ezek a `GameScreen`-ben találhatók).

Paraméterek

this	A `Carcassone` struktúra.

4.1.3.22 Carcassone__Game__move_board()

```
void Carcassone__Game__move_board (
            Carcassone * this,
            float dt )
```

Játéktábla mozgatása.

A nyilak segítségével a játéktábla látható részét mozgatja.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza az SDL kontextust.
dt	Deltaidő.

4.1.3.23 Carcassone__Game__render()

```
void Carcassone__Game__render (
            Carcassone * this )
```

Játéknézet megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amihez a nézet tartozik.

4.1.3.24 Carcassone__Game__render_board()

```
void Carcassone__Game__render_board (
            Carcassone * this )
```

Játéktábla renderelése.

Paraméterek

this	A Carcassone struktúra, ami tartalmazza a renderert.

4.1.3.25 Carcassone__Game__render_drawn_tile()

```
void Carcassone__Game__render_drawn_tile (
            Carcassone * this )
```

A pakli tetején levő kártya és a pakli méretének renderelése.

Paraméterek

this	A Carcassone struktúra, ami tartalmazza a renderert.

4.1.3.26 Carcassone__Game__render_game_over()

```
void Carcassone__Game__render_game_over (
            Carcassone * this )
```

A "game over" állapot megjelenítése.

Megjeleníti a koronát a nyertes játékos stat panelje felett.

Paraméterek

this	A Carcassone struktúra, ami tartalmazza a renderert.

4.1.3.27 Carcassone__Game__render_meeples()

```
void Carcassone__Game__render_meeples (
            Carcassone * this )
```

A lehelyezett alattvalók renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.

4.1.3.28 Carcassone__Game__render_player_stats()

```
void Carcassone__Game__render_player_stats (
            Carcassone * this )
```

A splash cím renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza az SDL kontextust.

4.1.3.29 Carcassone__Game__wrapup()

```
void Carcassone__Game__wrapup (
            Carcassone * this,
            bool concede )
```

"Game over" szituáció menedzselése.

Nyertes kiválasztása, dicsőséglista frissítése és visszatérés a menübe.

Paraméterek

this	A `Carcassone` struktúra, amihez a a nézet tartozik.
concede	Feladás történt e.

4.1.3.30 Carcassone__Lboard__construct()

```
void Carcassone__Lboard__construct (
            Carcassone * this )
```

Létrehozza a dicsőséglistanézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Lboard__destroy` függvényt.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a dicsőséglistanézetét.

4.1.3.31 Carcassone__Lboard__destroy()

```
void Carcassone__Lboard__destroy (
            Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúrához tartozó `LeaderboardScreen` által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.32 Carcassone__Lboard__handle_input()

```
void Carcassone__Lboard__handle_input (
            Carcassone * this )
```

Inputok kezelése dicsőglistanézetben.

Paraméterek

this	A `Carcassone` struktúra, amihez tartozik a dicsőglistanézet.

4.1.3.33 Carcassone__Lboard__init_list_texture()

```
void Carcassone__Lboard__init_list_texture (
            Carcassone * this )
```

Létrehozza a dicsőséglistanézethez a rekordokat.

Akkor kell meghívni, ha frissül a rekordfájl.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a dicsőséglistanézetét.

4.1.3.34 Carcassone__Lboard__render()

```
void Carcassone__Lboard__render (
            Carcassone * this )
```

Dicsőséglistanézet megjelenítése.

Paraméterek

this	A Carcassone struktúra, amihez a dicsőséglista tartozik.

4.1.3.35 Carcassone__Menu__construct()

```
void Carcassone__Menu__construct (
            Carcassone * this )
```

Létrehozza a menünézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a Carcassone__Menu__destroy függvényt.

Paraméterek

this	A Carcassone struktúra, amelynek létrehozza a menünézetét.

4.1.3.36 Carcassone__Menu__destroy()

```
void Carcassone__Menu__destroy (
            Carcassone * this )
```

Felszabadítja a megadott Carcassone struktúrához tartozó MenuScreen által lefoglalt memóriát.

Paraméterek

this	A Carcassone struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.37 Carcassone__Menu__handle_input()

```
void Carcassone__Menu__handle_input (
            Carcassone * this )
```

Inputok kezelése menünézetben.

Paraméterek

this	A `Carcassone` struktúra, amihez tartozik a menünézet.

4.1.3.38 Carcassone__Menu__render()

```
void Carcassone__Menu__render (
            Carcassone * this )
```

Menünézet megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amihez a menü tartozik.

4.1.3.39 Carcassone__Prompt__construct()

```
Prompt Carcassone__Prompt__construct (
            Carcassone * this,
            TTF_Font * font,
            char * default_label,
            SDL_Rect global_rect,
            SDL_Color bg_color,
            SDL_Color text_color )
```

Létrehoz egy szöveginputot.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Prompt__destroy` függvényt.

Paraméterek

this	A renderert tartalmazó `Carcassone` struktúra.
font	A címkéhez használt betűtípus.
default_label	A címke tartalma.
global_rect	A gomb globális pozíciója (az ablakon).
bg_color	A gomb színe.
text_color	A címke színe.

Visszatérési érték

Az új `Prompt`.

4.1.3.40 Carcassone__Prompt__destroy()

```
void Carcassone__Prompt__destroy (
            Carcassone * this,
            Prompt * prompt )
```

Felszabadítja a megadott `Prompt` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek az ablakára az adott szöveginput renderelve van (nem használt, régi verzió miatt).
prompt	A `Prompt` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.1.3.41 Carcassone__Prompt__edit()

```
void Carcassone__Prompt__edit (
            Carcassone * this,
            Prompt * prompt,
            char * new_label,
            bool concat )
```

Vagy hozzátold a szöveginput címkéjéhez, vagy lecseréli azt.

Paraméterek

this	A renderert tartalmazó `Carcassone` struktúra.
prompt	A szöveginput.
new_label	Az új címke vagy a hozzátoldott szöveg.
concat	Hozzátoldás e.

4.1.3.42 Carcassone__Prompt__render()

```
void Carcassone__Prompt__render (
            Carcassone * this,
            Prompt * prompt )
```

A szöveginput megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amelyhez a renderer tartozik.
prompt	A megjelenítendő szöveginput.

4.1.3.43 Carcassone__render_splash_title()

```
void Carcassone__render_splash_title (
            Carcassone * this,
            SDL_Rect * dst_rect )
```

A splash cím renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.
dst_rect	A splash cím pozíciója.

4.1.3.44 Carcassone__run()

```
void Carcassone__run (
            Carcassone * this )
```

A fő programciklus.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza az SDL kontextust.

4.1.3.45 Carcassone__switch_state()

```
void Carcassone__switch_state (
            Carcassone * this,
            AppState new_state )
```

Megváltoztatja a program nézetét.

Nem csinál semmit, ha a régi és új nézet megegyezik.

Paraméterek

this	A `Carcassone` struktúra aminek a nézetét meg kell változtatni.
new_state	Az új nézet.

4.2 app.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_APP_H
2 #define CRCLONE_APP_H
3
4 #include <bits/pthreadtypes.h>
5 #include <stdbool.h>
6 #include <SDL2/SDL.h>
7 #include <SDL2/SDL_ttf.h>
8
9 #include "game/tile.h"
10 #include "game/player.h"
11 #include "ui.h"
12
14 #define FPS 60
16 #define MIN(x, y) ((x < y) ?  x :  y)
18 #define COLOR_BLUE 25, 30, 91, 255
19 #define COLOR_LIGHTBLUE 153, 204, 255, 255
20 #define COLOR_WHITE 255, 255, 255, 255
21 #define COLOR_RED 255, 0, 0, 255
22 #define COLOR_SALMON 255, 145, 164, 255
23 #define COLOR_BG 102, 102, 153, 255
25 #define DBG_LOG(x, ...)  SDL_LogDebug(SDL_LOG_CATEGORY_CUSTOM, x, ##__VA_ARGS__)
26
28 typedef enum {
29     MENU,
30     GAME,
31     LEADERBOARD
32 } AppState;
33
35 typedef struct {
36     // Ablak dimenziói.
37     int width, height;
38
39     // Fut e.
40     bool is_running;
41
42     // Az ablak.
43     SDL_Window* window;
44
45     // Ablakikon.
46     SDL_Surface* window_icon;
47
48     // Splash cím.
49     SDL_Texture* splash_title;
50
51     // A renderer.
52     SDL_Renderer* renderer;
53
54     // A használt betűtípusok.
55     TTF_Font* default_font, * small_font;
56
57     // Állapotok.
58     AppState state;
59     MenuScreen* menu_screen;
60     LeaderboardScreen* lboard_screen;
61     GameScreen* game_screen;
62
63     // Mutex arra az esetre, mikor visszamegyünk a menübe game over-t követve.
64     SDL_mutex* smutex;
65 } Carcassone;
66 Carcassone* Carcassone__construct(int, int, char const*);
67 void Carcassone__destroy(Carcassone*);
68 void Carcassone__switch_state(Carcassone*, AppState);
69 void Carcassone__run(Carcassone*);
70 void Carcassone__render_splash_title(Carcassone*, SDL_Rect*);
71
72 void Carcassone__Menu__construct(Carcassone*);
73 void Carcassone__Menu__destroy(Carcassone*);
74 void Carcassone__Menu__render(Carcassone*);
75 void Carcassone__Menu__handle_input(Carcassone*);
76
77 void Carcassone__Lboard__construct(Carcassone*);
78 void Carcassone__Lboard__destroy(Carcassone*);
79 void Carcassone__Lboard__render(Carcassone*);
80 void Carcassone__Lboard__init_list_texture(Carcassone*);
81 void Carcassone__Lboard__handle_input(Carcassone*);
82
83 void Carcassone__Game__construct(Carcassone*);
84 void Carcassone__Game__destroy(Carcassone*);
85 void Carcassone__Game__handle_input(Carcassone*, float);
86 void Carcassone__Game__init_players(Carcassone*);
87 void Carcassone__Game__init_pile(Carcassone*);
88 void Carcassone__Game__init_board(Carcassone*);
89 void Carcassone__Game__init_counter(Carcassone*);
90 void Carcassone__Game__render(Carcassone*);
91 void Carcassone__Game__render_board(Carcassone*);
92 void Carcassone__Game__render_drawn_tile(Carcassone*);
93 void Carcassone__Game__render_player_stats(Carcassone*);
```

```
94 void Carcassone__Game__render_meeples(Carcassone*);
95 void Carcassone__Game__render_game_over(Carcassone*);
96 void Carcassone__Game__wrapup(Carcassone*, bool);
97 void Carcassone__Game__move_board(Carcassone*, float);
98 bool Carcassone__Game__draw_new(Carcassone*);
99 bool Carcassone__Game__check_names_valid(Carcassone*);
100 void Carcassone__Game__indicate_possible_placements(Carcassone*);
101 bool Carcassone__Game__check_surrounding_tiles(Carcassone*, SDL_Point);
102 void Carcassone__Game__calculate_points(Carcassone*);
103 bool Carcassone__Game__check_if_possible(Carcassone*);
104 void Carcassone__Game__calculate_scores_for_cloister(Carcassone*, Tile*);
105 void Carcassone__Game__calculate_scores_for_graph(Carcassone*, Tile*, ConnectionType);
106
107 Button Carcassone__Button__construct(Carcassone*, TTF_Font*, char*, SDL_Rect, SDL_Color, SDL_Color,
      bool);
108 void Carcassone__Button__destroy(Carcassone*, Button*);
109 void Carcassone__Button__render(Carcassone*, Button*);
110 bool Carcassone__Button__hover(Carcassone*, Button*, SDL_Point);
111
112 Prompt Carcassone__Prompt__construct(Carcassone*, TTF_Font*, char*, SDL_Rect, SDL_Color, SDL_Color);
113 void Carcassone__Prompt__destroy(Carcassone*, Prompt*);
114 void Carcassone__Prompt__render(Carcassone*, Prompt*);
115 void Carcassone__Prompt__edit(Carcassone*, Prompt*, char*, bool);
116
117 #endif
```

4.3 include/game/meeple.h fájlreferencia

```
#include <stdbool.h>
#include <SDL2/SDL.h>
```

Adatszerkezetek

- struct Meeple

Függvények

- Meeple Meeple__construct (SDL_Texture ∗)

  Létrehoz egy alattvalót.
- void Meeple__render (SDL_Renderer ∗, unsigned int, SDL_Color)
- void Meeple__destroy (Meeple ∗)

4.3.1 Függvények dokumentációja

4.3.1.1 Meeple__construct()

```
Meeple Meeple__construct (
            SDL_Texture * texture )
```

Létrehoz egy alattvalót.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Meeple__destroy` függvényt.

Paraméterek

texture	A kinézete.

Visszatérési érték

Az új `Meeple` struktúra.

4.3.1.2 Meeple__destroy()

```
void Meeple__destroy (
            Meeple *  )
```

4.3.1.3 Meeple__render()

```
void Meeple__render (
            SDL_Renderer * ,
            unsigned int ,
            SDL_Color  )
```

4.4 meeple.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_MEEPLE_H
2 #define CRCLONE_MEEPLE_H
3
4 #include <stdbool.h>
5 #include <SDL2/SDL.h>
6
8 typedef struct {
9     // Le van e helyezve.
10     bool is_placed;
11
12     // Táblakoordináták.
13     // Ha !is_placed, akkor akár invalid, "szemét" értéket is tárolhat.
14     int x, y;
15
16     // Kinézete.  (játékosoként változtatható).
17     SDL_Texture* texture;
18 } Meeple;
19 Meeple Meeple__construct(SDL_Texture*);
20 void Meeple__render(SDL_Renderer*, unsigned int, SDL_Color);
21 void Meeple__destroy(Meeple*);
22
23 #endif
```

4.5 include/game/player.h fájlreferencia

```
#include <stdlib.h>
#include <stdbool.h>
#include <wchar.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "game/meeple.h"
```

Adatszerkezetek

- struct Player
- struct LeaderboardEntry
- struct Leaderboard

Makródefiníciók

- #define MAX_MEEPLES 7
- #define MAX_PLAYER_NAME_LEN 24

Függvények

- Player Player__construct (SDL_Renderer ∗, TTF_Font ∗, char ∗, char const ∗)

  Létrehoz egy játékost.
- void Player__place_meeple (Player ∗, SDL_Point)

  Alattvaló lehelyezése.
- void Player__reclaim_meeple (Player ∗, Meeple ∗)

  Lehelyezett alattvaló visszaszerzése.
- void Player__render (Player ∗, SDL_Renderer ∗, TTF_Font ∗)

  A megadott játékos statisztájának lerenderelése.
- void Player__add_to_score (Player ∗, unsigned int)

  Játékos pontszámának növelése.
- void Player__destroy (Player ∗)

  Felszabadítja a megadott `Player` struktúra által lefoglalt memóriát.
- Leaderboard ∗ Leaderboard__construct (char const ∗)

  Létrehozza a dicsőséglistát.
- void Leaderboard__destroy (Leaderboard ∗)

  Felszabadítja a megadott `Leaderboard` struktúra által lefoglalt memóriát.
- void Leaderboard__sort (Leaderboard ∗)

  A beolvasott rekordok csökkenő sorba rendezése.
- bool Leaderboard__load (Leaderboard ∗)

  Betölti a rekordfájl tartalmát.
- bool Leaderboard__insert_new (Leaderboard ∗, Player ∗)

  Új rekord beszúrása és mentése.

4.5.1 Makródefiníciók dokumentációja

4.5.1.1 MAX_MEEPLES

```
#define MAX_MEEPLES 7
```

Maximum alattvalók száma per játékos.

4.5.1.2 MAX_PLAYER_NAME_LEN

```
#define MAX_PLAYER_NAME_LEN 24
```

Egy játékos nevének max hossza (multibyte és nullterminálo nélkül).

4.5.2 Függvények dokumentációja

4.5.2.1 Leaderboard__construct()

```
Leaderboard * Leaderboard__construct (
            char const * records_file_path )
```

Létrehozza a dicsőséglistát.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Leaderboard__destroy` függvényt.

Paraméterek

records_file_path	A fájl elérési útvonala, ahonnan beolvassa a játékosneveket és rekordokat.

Visszatérési érték

Pointer az újonnan létrehozott `Leaderboard`-ra.

4.5.2.2 Leaderboard__destroy()

```
void Leaderboard__destroy (
            Leaderboard * this )
```

Felszabadítja a megadott `Leaderboard` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Leaderboard` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.5.2.3 Leaderboard__insert_new()

```
bool Leaderboard__insert_new (
            Leaderboard * this,
            Player * new )
```

Új rekord beszúrása és mentése.

Megjegyzés: A visszatérési érték nem feltétlen van kihasználva, elhanyagolható.

Paraméterek

this	A `Leaderboard` struktúra, amihez tartozó rekordfájlba történjen a mentés.
new	A potenciálisan új, rekordot elérő játékos.

Visszatérési érték

Már tartalmazta e a dicsőséglista a játékost.

4.5.2.4 Leaderboard__load()

```
bool Leaderboard__load (
            Leaderboard * this )
```

Betölti a rekordfájl tartalmát.

Érdemes meghívni a `Leaderboard__sort`-ot ezután.

Paraméterek

this	A `Leaderboard` struktúra, amelyhez már tartozik egy rekordfájl.

Visszatérési érték

Hibamentesen futott e le.

4.5.2.5 Leaderboard__sort()

```
void Leaderboard__sort (
            Leaderboard * this )
```

A beolvasott rekordok csökkenő sorba rendezése.

Paraméterek

this	A `Leaderboard` struktúra, amely a beolvasott rekordokat tárolja.

4.5.2.6 Player__add_to_score()

```
void Player__add_to_score (
            Player * this,
            unsigned int add )
```

Játékos pontszámának növelése.

Beállítja az `update_score`-t, a következő render esetén frissíti a számlálót.

Paraméterek

this	A `Player`, amelynek a pontszámát növeljük.
add	A hozzáadott mennyiség.

4.5.2.7 Player__construct()

```
Player Player__construct (
            SDL_Renderer * renderer,
            TTF_Font * font,
            char * name,
            char const * meeple_outfit )
```

Létrehoz egy játékost.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Player__destroy` függvényt.

Paraméterek

renderer	A renderer amivel a főablakra rajzolunk.
font	A statisztikák kiírásához használt betűtípus.
name	A neve.
meeple_outfit	Az alattvalóinak kinézete (képfájl).

Visszatérési érték

Az új `Player`.

4.5.2.8 Player__destroy()

```
void Player__destroy (
            Player * this )
```

Felszabadítja a megadott `Player` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Player` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.5.2.9 Player__place_meeple()

```
void Player__place_meeple (
            Player * this,
            SDL_Point tile_index )
```

Alattvaló lehelyezése.

Nem viszgálja meg, hogy a mezőkártya valóban megfelelő e.

Paraméterek

this	Az a `Player`, amelynek egy alattvalóját le kell helyezni.
tile_index	A mezőkártya koordinátája a játéktáblán, amire az alattvaló le lesz téve.

4.5.2.10 Player__reclaim_meeple()

```
void Player__reclaim_meeple (
            Player * this,
            Meeple * to_reclaim )
```

Lehelyezett alattvaló visszaszerzése.

Paraméterek

this	A `Player` struktúra, amelynek egy alattvalóját vissza kell szerezni.
to_reclaim	A visszaszerzendő `Meeple`.

4.5.2.11 Player__render()

```
void Player__render (
            Player * this,
            SDL_Renderer * renderer,
            TTF_Font * font )
```

A megadott játékos statisztájának lerenderelése.

Paraméterek

this	A `Player` struktúra, aminek a statisztikáját akarjuk megjeleníteni.
renderer	A renderer amivel a főablakra rajzolunk.
font	A statisztikák kiírásához használt betűtípus.

4.6 player.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_PLAYER_H
2 #define CRCLONE_PLAYER_H
3
4 #include <stdlib.h>
5 #include <stdbool.h>
6 #include <wchar.h>
7 #include <SDL2/SDL.h>
8 #include <SDL2/SDL_ttf.h>
9
10 #include "game/meeple.h"
11
13 #define MAX_MEEPLES 7
15 #define MAX_PLAYER_NAME_LEN 24
16
17 typedef struct {
18     // Általános adatok.
19     // UTF-8 support:  sajnos a TTF_RenderUTF8_Blended valójában char*-t fogad el, ezért nem tárolhatom
      wchar_t*-ben.
20     char name[(MAX_PLAYER_NAME_LEN + 1) * sizeof(wchar_t)];
21     unsigned int score;
22
23     // Tett e már le kártyát az adott körben.
24     bool has_placed_card;
25
26     // Alattvalók.
27     Meeple meeples[MAX_MEEPLES];
28     size_t meeples_at_hand;
29
30     // Textúrák.
31     SDL_Texture* score_counter;
32     SDL_Texture* handle_texture;
33     SDL_Texture* stat_panel;
34     SDL_Texture* own_meeple_texture;
35
36     // Kell e frissíteni a pontot.
37     bool update_score;
38 } Player;
39 Player Player__construct(SDL_Renderer*, TTF_Font*, char*, char const*);
40 void Player__place_meeple(Player*, SDL_Point);
41 void Player__reclaim_meeple(Player*, Meeple*);
42 void Player__render(Player*, SDL_Renderer*, TTF_Font*);
43 void Player__add_to_score(Player*, unsigned int);
44 void Player__destroy(Player*);
45
47 typedef struct {
48     // Játékos neve.
49     char name[(MAX_PLAYER_NAME_LEN + 1) * sizeof(wchar_t)];
50
51     // Játékos rekordja.
52     unsigned int highscore;
53 } LeaderboardEntry;
54
56 typedef struct {
57     // A rekordfájl elérési útvonala.
58     char const* records_file_path;
59
60     // A rekordok.
61     LeaderboardEntry* entries;
62     size_t entries_size;
63 } Leaderboard;
64 Leaderboard* Leaderboard__construct(char const*);
65 void Leaderboard__destroy(Leaderboard*);
66 void Leaderboard__sort(Leaderboard*);
67 bool Leaderboard__load(Leaderboard*);
68 bool Leaderboard__insert_new(Leaderboard*, Player*);
69
70 #endif
```

4.7 include/game/tile.h fájlreferencia

```
#include <stdbool.h>
#include <SDL2/SDL.h>
```

Adatszerkezetek

- struct Tile
- struct CardPile
- struct TilesetWrapper

Makródefiníciók

- #define TILE_SIZE 100
- #define TILE_SIZE_SRC 64
- #define PILE_SIZE 71

Típusdefiníciók

- typedef struct CardPile CardPile

Enumerációk

- enum ConnectionType { NONE = 0U , FIELD , ROAD , CASTLE }
- enum ConnectionDirection { NORTH = 0U , EAST = 1U , SOUTH = 2U , WEST = 3U }
- enum TileType {
  EMPTY = -1 , FIELD_CLOISTER_ROAD_S = 0 , FIELD_CLOISTER_ROAD_NS , FIELD_VILLAGE_ROAD_S
  ,
  FIELD_VILLAGE_ROAD_NS , ROAD_NS , ROAD_NW , ROAD_NWE ,
  ROAD_NSWE , CASTLE_TOWN , CASTLE_PANTHEON , CASTLE_TUNNEL ,
  CASTLE_CORNER_WALL , CASTLE_CORNER_WALL_ROAD_BY , CASTLE_CAP_WALL , CASTLE_CAP_WALL_ROAD_TO
  ,
  CASTLE_CAP_WALL_ROAD_BY , CASTLE_SHIRT_WALL , CASTLE_SHIRT_WALL_ROAD_TO ,
  TILETYPE_SIZE__ }

Függvények

- void Tile__construct (Tile *, TileType, SDL_Point, SDL_FPoint)

  Létrehoz egy mezőkártyát.
- bool Tile__point_in_tile (Tile *, SDL_FPoint)

  Ellenőrzi hogy egy pont az adott kártya területében van e.
- void Tile__move_by (Tile *, float, float)

  Elmozgatja a megadott mezőkártyát egy adott irányba.
- void Tile__rotate (Tile *)

  Órajárásával megegyező irányába elforgatja 90 fokkal a megadott mezőkártyát.
- void Tile__set_rotation (Tile *, unsigned short)

  A megadott mezőkártyának elforgatását beállítja pontosan a kapott értékre.
- void Tile__set_type (Tile *, TileType, unsigned short)

Adott mezőkártya típusának megváltoztatása.

- CardPile * CardPile__construct (void)

  Létrehozza a kártyapaklit.
- CardPile * CardPile__pop (CardPile *, TileType *)

  Leveszi és eltárolja a pakli legfelső elemét.
- CardPile * CardPile__push (CardPile *, TileType)

  A megadott paklira rátesz egy új kártyát.
- void CardPile__destroy (CardPile *)

  Felszabadítja a megadott `CardPile` strúktúrát (a legutolsó elemen kell meghívni).
- TilesetWrapper TilesetWrapper__construct (SDL_Renderer *)

  Létrehoz egy új `TilesetWrapper`-t.
- void TilesetWrapper__destroy (TilesetWrapper *)

  Felszabadítja a megadott `TilesetWrapper` struktúrában tárolt textúra atlaszt.
- SDL_Rect get_texture_rect_for (TileType)

  A megadott kártyatípushoz tartozó textúra pozícióját számolja ki az atlaszon.

4.7.1 Makródefiníciók dokumentációja

4.7.1.1 PILE_SIZE

```
#define PILE_SIZE 71
```

A pakli maximális mérete.

4.7.1.2 TILE_SIZE

```
#define TILE_SIZE 100
```

Megjelenő mezőkártyák mérete.

4.7.1.3 TILE_SIZE_SRC

```
#define TILE_SIZE_SRC 64
```

A texture atlas-ban egy mezőkártya mérete.

4.7.2 Típusdefiníciók dokumentációja

4.7.2.1 CardPile

```
typedef struct CardPile CardPile
```

Kártyapakli stackként implementálva.

4.7.3 Enumerációk dokumentációja

4.7.3.1 ConnectionDirection

```
enum ConnectionDirection
```

Alias-ok a kártyaoldalaknak.

Enumeráció-értékek

NORTH	
EAST	
SOUTH	
WEST	

4.7.3.2 ConnectionType

```
enum ConnectionType
```

Kártyaoldal típusok.

Enumeráció-értékek

NONE	
FIELD	
ROAD	
CASTLE	

4.7.3.3 TileType

```
enum TileType
```

Kártyatípusok.

Enumeráció-értékek

EMPTY	
FIELD_CLOISTER_ROAD_S	
FIELD_CLOISTER_ROAD_NS	
FIELD_VILLAGE_ROAD_S	
FIELD_VILLAGE_ROAD_NS	
ROAD_NS	

Enumeráció-értékek

ROAD_NW	
ROAD_NWE	
ROAD_NSWE	
CASTLE_TOWN	
CASTLE_PANTHEON	
CASTLE_TUNNEL	
CASTLE_CORNER_WALL	
CASTLE_CORNER_WALL_ROAD_BY	
CASTLE_CAP_WALL	
CASTLE_CAP_WALL_ROAD_TO	
CASTLE_CAP_WALL_ROAD_BY	
CASTLE_SHIRT_WALL	
CASTLE_SHIRT_WALL_ROAD_TO	
TILETYPE_SIZE__	

4.7.4 Függvények dokumentációja

4.7.4.1 CardPile__construct()

```
CardPile * CardPile__construct (
            void  )
```

Létrehozza a kártyapaklit.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `CardPile__destroy` függvényt.

Visszatérési érték

Pointer az új `CardPile` legelső elemére (mindig EMPTY típust tárol) vagy NULL ha hiba volt.

4.7.4.2 CardPile__destroy()

```
void CardPile__destroy (
            CardPile * this )
```

Felszabadítja a megadott `CardPile` strúktúrát (a legutolsó elemen kell meghívni).

Rekúrzívan működik.

Paraméterek

this	A `CardPile`, aminek a lefoglalt memóriáját fel kell szabadítani.

4.7.4.3 CardPile__pop()

```
CardPile * CardPile__pop (
            CardPile * this,
            TileType * popped )
```

Leveszi és eltárolja a pakli legfelső elemét.

Paraméterek

this	A pakli aminek leveszi a legfelső elemét.
popped	Ahova eltárolja a kapott kártyatípust.

Visszatérési érték

Pointer a `CardPile` új legfelső elemére vagy NULL ha hiba volt.

4.7.4.4 CardPile__push()

```
CardPile * CardPile__push (
            CardPile * this,
            TileType new_type )
```

A megadott paklira rátesz egy új kártyát.

Paraméterek

this	A pakli amire rá kell tenni egy új kártyát.
new_type	A tárolandó kártyatípus.

Visszatérési érték

Pointer az új legfelső elemre vagy NULL ha hiba volt.

4.7.4.5 get_texture_rect_for()

```
SDL_Rect get_texture_rect_for (
            TileType type )
```

A megadott kártyatípushoz tartozó textúra pozícióját számolja ki az atlaszon.

Paraméterek

type	A mezőkártya típusa, amelynek a textúráját szeretnénk.

Visszatérési érték

A textúra pozíciója az atlaszon.

4.7.4.6 Tile__construct()

```
void Tile__construct (
            Tile * this,
            TileType type,
            SDL_Point board_coords,
            SDL_FPoint offset )
```

Létrehoz egy mezőkártyát.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Tile__destroy` függvényt.

Paraméterek

this	A létrejövő kártya.
type	A kártya típusa.
board_coords	A cella koordinátái a táblán (0 és BOARD_SIZE közt).
offset	A tábla koordinátái.

4.7.4.7 Tile__move_by()

```
void Tile__move_by (
            Tile * this,
            float mvx,
            float mvy )
```

Elmozgatja a megadott mezőkártyát egy adott irányba.

Paraméterek

this	Az adott `Tile`.
mvx	Mozgatás az x tengelyen.
mvy	Mozgatás az y tengelyen.

4.7.4.8 Tile__point_in_tile()

```
bool Tile__point_in_tile (
            Tile * this,
            SDL_FPoint pt )
```

Ellenőrzi hogy egy pont az adott kártya területében van e.

Paraméterek

this	A vizsgálandó mezőkártya.
pt	A vizsgálandó pont.

Visszatérési érték

Benne van e a pont.

4.7.4.9 Tile__rotate()

```
void Tile__rotate (
            Tile * this )
```

Órajárásával megegyező irányába elforgatja 90 fokkal a megadott mezőkártyát.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.

4.7.4.10 Tile__set_rotation()

```
void Tile__set_rotation (
            Tile * this,
            unsigned short new_rotation )
```

A megadott mezőkártyának elforgatását beállítja pontosan a kapott értékre.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.
new_rotation	Az elforgatás értéke.

4.7.4.11 Tile__set_type()

```
void Tile__set_type (
            Tile * this,
            TileType new_type,
            unsigned short new_rotation )
```

Adott mezőkártya típusának megváltoztatása.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.
new_type	Az új típusa.
new_rotation	-

4.7.4.12 TilesetWrapper__construct()

```
TilesetWrapper TilesetWrapper__construct (
            SDL_Renderer * renderer )
```

Létrehoz egy új `TilesetWrapper`-t.

Singletonként kell használni a programban. Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `TilesetWrapper__destroy` függvényt.

Paraméterek

renderer	A renderer amivel a főablakra rajzolunk.

Visszatérési érték

Az új atlasz wrapper.

4.7.4.13 TilesetWrapper__destroy()

```
void TilesetWrapper__destroy (
            TilesetWrapper * this )
```

Felszabadítja a megadott `TilesetWrapper` struktúrában tárolt textúra atlaszt.

Paraméterek

this	A `TilesetWrapper`, aminek a lefoglalt memóriáját fel kell szabadítani.

4.8 tile.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_TILE_H
2 #define CRCLONE_TILE_H
3
4 #include <stdbool.h>
5 #include <SDL2/SDL.h>
6
8 #define TILE_SIZE 100
10 #define TILE_SIZE_SRC 64
12 #define PILE_SIZE 71
13
15 typedef enum {
16     NONE = 0U,
17     FIELD,
18     ROAD,
19     CASTLE
20 } ConnectionType;
21
23 typedef enum {
24     NORTH = 0U,
25     EAST = 1U,
26     SOUTH = 2U,
27     WEST = 3U
28 } ConnectionDirection;
29
31 typedef enum {
32     EMPTY = -1,
33     FIELD_CLOISTER_ROAD_S = 0,
34     FIELD_CLOISTER_ROAD_NS,
35     FIELD_VILLAGE_ROAD_S,
36     FIELD_VILLAGE_ROAD_NS,
37     ROAD_NS,
38     ROAD_NW,
39     ROAD_NWE,
40     ROAD_NSWE,
41     CASTLE_TOWN,
42     CASTLE_PANTHEON,
43     CASTLE_TUNNEL,
44     CASTLE_CORNER_WALL,
45     CASTLE_CORNER_WALL_ROAD_BY,
46     CASTLE_CAP_WALL,
47     CASTLE_CAP_WALL_ROAD_TO,
48     CASTLE_CAP_WALL_ROAD_BY,
49     CASTLE_SHIRT_WALL,
50     CASTLE_SHIRT_WALL_ROAD_TO,
51     TILETYPE_SIZE__ // = 19, trick
52 } TileType;
53
55 typedef struct {
56     // Típusa.
57     TileType type;
58
59     // A négy oldal:  north, east, south, west.  (ehhez van a ConnectionDirection)
60     ConnectionType connections[4];
61
62     // Bal felső sarok koordinátái.
63     SDL_FPoint local_coords, global_coords;
64
65     // Táblán a koordináta.
66     SDL_Point board_coords;
67
68     // Elforgatás.
69     // `unsigned int`, mert csak egész foknyi forgatás van engedve (sőt, gyakorlatban az is csak 90 fok).
70     unsigned short rotation;
71
72     // Egyes kártyákat, amelyeknek minden oldalkapcsolata azonos, nem lehet forgatni.
73     bool rotatable;
74
75     // Volt e már pontozva.
76     bool is_scored;
77
78     // Lehet e tenni még rá alattvalót.  (kolostornál számít)
```

```
79     bool is_expired;
80 } Tile;
81 void Tile__construct(Tile*, TileType, SDL_Point, SDL_FPoint);
82 bool Tile__point_in_tile(Tile*, SDL_FPoint);
83 void Tile__move_by(Tile*, float, float);
84 void Tile__rotate(Tile*);
85 void Tile__set_rotation(Tile*, unsigned short);
86 void Tile__set_type(Tile*, TileType, unsigned short);
87
89 typedef struct CardPile {
90     // A tárolt kártyatípus.
91     TileType card;
92
93     // Az "alatta" levő kártya.
94     struct CardPile* next;
95 } CardPile;
96 CardPile* CardPile__construct(void);
97 CardPile* CardPile__pop(CardPile*, TileType*);
98 CardPile* CardPile__push(CardPile*, TileType);
99 void CardPile__destroy(CardPile*);
100
102 typedef struct {
103     // A tárolt atlasz.
104     SDL_Texture* tile_set;
105 } TilesetWrapper;
106 TilesetWrapper TilesetWrapper__construct(SDL_Renderer*);
107 void TilesetWrapper__destroy(TilesetWrapper*);
108 SDL_Rect get_texture_rect_for(TileType);
109
110 #endif
```

4.9 include/ui.h fájlreferencia

```
#include <stdbool.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "game/tile.h"
#include "game/player.h"
```

Adatszerkezetek

- struct Button
- struct Prompt
- struct MenuScreen
- struct LeaderboardScreen
- struct GameScreen

Makródefiníciók

- #define BOARD_SIZE 71

4.9.1 Makródefiníciók dokumentációja

4.9.1.1 BOARD_SIZE

```
#define BOARD_SIZE 71
```

A játéktábla mérete (BOARD_SIZE ∗ BOARD_SIZE)

4.10 ui.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_UI_H
2 #define CRCLONE_UI_H
3
4 #include <stdbool.h>
5 #include <SDL2/SDL.h>
6 #include <SDL2/SDL_ttf.h>
7
8 #include "game/tile.h"
9 #include "game/player.h"
10
12 #define BOARD_SIZE 71
13
15 typedef struct {
16     // Címke.
17     char* label;
18     SDL_Texture* label_texture;
19     SDL_Rect label_rect;
20
21     // Címke betűtípusa.
22     TTF_Font* used_font;
23
24     // A gomb pozíciója (lokálisan és globálisan).
25     SDL_Rect local_rect, global_rect;
26
27     // Háttérszín.
28     SDL_Color bg_color;
29 } Button;
30
32 typedef struct {
33     Button prompt;
34 } Prompt;
35
37 typedef struct {
38     // Háttérkép.
39     SDL_Texture* background;
40
41     // A "menü menü."
42     SDL_Rect button_container;
43     Button start_button;
44     Button lboard_button;
45 } MenuScreen;
46
48 typedef struct {
49     // Maga a dicsésglista (és adatai).
50     Leaderboard* leaderboard;
51
52     // Hibaüzenet hibás formátum esetén.
53     char syntax_error_msg[128+1];
54
55     // A rekordok egy textúrán.
56     SDL_Texture* list_texture;
57
58     // "Vissza" gomb.
59     Button back_button;
60 } LeaderboardScreen;
61
63 typedef struct {
64     // A ready_button meg lett e nyomva.
65     bool is_ready;
66
67     // Vége van e játéknak.
68     bool is_game_over;
69
70     // "Global state" a lenyomott nyílgombokhoz.  (a tábla smooth mozgatása miatt)
71     int held_arrow_keys[4];
72
73     // A tábla felsősarkának koordinátái.
74     SDL_FPoint board_offset;
75
76     // Tábla.
77     Tile** board;
78     SDL_Texture* board_texture;
79
80     // Húzott kártya.
81     Tile* drawn_tile;
82
83     // Pakli.
84     CardPile* card_pile;
85     size_t pile_index;
86
87     // Pakliszámláló.
88     SDL_Texture* pile_counter;
```

```
89
90      // Kell e frissíteni a pakliszámlálót.
91      bool update_counter;
92
93      // Játékosok, referenciák.
94      Player players[2];
95      Player* curr_player;
96      Player* winner;
97
98      // A korona textúra.
99      SDL_Texture* crown_texture;
100
101      // Játékosneveknek a szöveginputjai.
102      SDL_Texture* player_input_labels[2];
103      Prompt player_name_inputs[2];
104      Prompt* active_input;
105
106      // "Ok" gomb, "Kör vége" gomb, "Felad" gomb.
107      Button ready_button, end_turn_button, concede_button;
108
109      // Az táblaatlasz.
110      TilesetWrapper tileset_wrapper;
111  } GameScreen;
112
113  #endif
```

4.11 include/utils.h fájlreferencia

```
#include <stdbool.h>
#include <SDL2/SDL_render.h>
```

Függvények

- size_t mb_strlen (char const *)

  Multibyte (akár UTF8) karaktereket tartalmazó sztring "látható karaktereinek" számának meghatározása.
- char * strdup_ (char const *)

  `strdup` *POSIX függvényhez hasonló.*
- char * strcatdyn (char *, char const *)

  Dinamikus sztringhez hozzátoldás.
- char * mb_remove_last_char_dyn (char *)

  Dinamikus sztring utolsó karakterének (multibyte) eltávolítása.
- void destroy_SDL_Texture (SDL_Texture *)

  `SDL_Texture` *felszabadítása, de kezeli azt is ha NULL a megadott textúra.*
- SDL_Texture * create_SDL_texture_from_BMP (SDL_Renderer *, char const *)

  `SDL_Texture` *létrehozása egy BMP képfájlból.*

4.11.1 Függvények dokumentációja

4.11.1.1 create_SDL_texture_from_BMP()

```
SDL_Texture * create_SDL_texture_from_BMP (
            SDL_Renderer * renderer,
            char const * source_path )
```

`SDL_Texture` létrehozása egy BMP képfájlból.

Megjegyzés: A létrejött textúra a `destroy_SDL_Texture`-rel törlendő.

Paraméterek

renderer	A főablakhoz tartozó renderer.
source_path	A BMP képfájl elérési útvonala.

Visszatérési érték

A létrehozott textúrára mutató pointer.

4.11.1.2 destroy_SDL_Texture()

```
void destroy_SDL_Texture (
            SDL_Texture * texture )
```

`SDL_Texture` felszabadítása, de kezeli azt is ha NULL a megadott textúra.

Paraméterek

texture	A törlendő textúra.

4.11.1.3 mb_remove_last_char_dyn()

```
char * mb_remove_last_char_dyn (
            char * str )
```

Dinamikus sztring utolsó karakterének (multibyte) eltávolítása.

Megjegyzés: Az eredeti sztringet felszabadítja.

Paraméterek

str	A dinamikusan foglalt sztring.

Visszatérési érték

Egy új dinamikusan foglalt sztring, amelynek tartalma az `str`-ével megegyezik, kivéve annak utolsó karaktere.

4.11.1.4 mb_strlen()

```
size_t mb_strlen (
            char const * str )
```

Multibyte (akár UTF8) karaktereket tartalmazó sztring "látható karaktereinek" számának meghatározása.

Paraméterek

str	A sztring aminek meghatározzuk a hosszát.

Visszatérési érték

A megadott sztring hossza.

4.11.1.5 strcatdyn()

```
char * strcatdyn (
            char * original,
            char const * to_cat )
```

Dinamikus sztringhez hozzátoldás.

Megjegyzés: Az eredeti sztringet felszabadítja.

Paraméterek

original	A dinamikusan foglalt sztring, amelyhez hozzátoldunk.
to_cat	A hozzátoldandó sztring (nem feltétlen dinamikus).

Visszatérési érték

Egy új dinamikusan foglalt sztring, amelynek tartalma az `original` és `to_cat`-é.

4.11.1.6 strdup_()

```
char * strdup_ (
            char const * str )
```

`strdup` POSIX függvényhez hasonló.

Paraméterek

str	A sztring amit dinamikusan le kell foglalni.

Visszatérési érték

Az új, dinamikusan foglalt sztring.

4.12 utils.h

Ugrás a fájl dokumentációjához.

```
1 #ifndef CRCLONE_UTILS_H
2 #define CRCLONE_UTILS_H
3
4 #include <stdbool.h>
5 #include <SDL2/SDL_render.h>
6
7 // Pár segédfüggvény.
8
9 size_t mb_strlen(char const*);
10 char* strdup_(char const*);
11 char* strcatdyn(char*, char const*);
12 char* mb_remove_last_char_dyn(char*);
13 void destroy_SDL_Texture(SDL_Texture*);
14 SDL_Texture* create_SDL_texture_from_BMP(SDL_Renderer*, char const*);
15
16 #endif
```

4.13 src/app.c fájlreferencia

```
#include <stdio.h>
#include <time.h>
#include <limits.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "utils.h"
#include "app.h"
```

Függvények

- Carcassone * Carcassone__construct (int width, int height, char const *title)

  Initicializálja az összes nézetet, SDL és TTF kontextusokat.
- void Carcassone__destroy (Carcassone *this)

  Felszabadítja a megadott `Carcassone` struktúra által lefoglalt memóriát.
- void Carcassone__switch_state (Carcassone *this, AppState new_state)

  Megváltoztatja a program nézetét.
- void Carcassone__render_splash_title (Carcassone *this, SDL_Rect *dst_rect)

  A splash cím renderelése.
- void Carcassone__run (Carcassone *this)

  A fő programciklus.

4.13.1 Függvények dokumentációja

4.13.1.1 Carcassone__construct()

```
Carcassone * Carcassone__construct (
             int width,
             int height,
             char const * title )
```

Initicializálja az összes nézetet, SDL és TTF kontextusokat.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__destroy` függvényt.

Paraméterek

width	Az ablak szélessége.
height	Az ablak magassága.
title	Az ablak címe.

Visszatérési érték

Pointer az újonnan létrehozott `Carcassone` structra.

4.13.1.2 Carcassone__destroy()

```
void Carcassone__destroy (
             Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.13.1.3 Carcassone__render_splash_title()

```
void Carcassone__render_splash_title (
             Carcassone * this,
             SDL_Rect * dst_rect )
```

A splash cím renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.
dst_rect	A splash cím pozíciója.

4.13.1.4 Carcassone__run()

```
void Carcassone__run (
            Carcassone * this )
```

A fő programciklus.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza az SDL kontextust.

4.13.1.5 Carcassone__switch_state()

```
void Carcassone__switch_state (
            Carcassone * this,
            AppState new_state )
```

Megváltoztatja a program nézetét.

Nem csinál semmit, ha a régi és új nézet megegyezik.

Paraméterek

this	A `Carcassone` struktúra aminek a nézetét meg kell változtatni.
new_state	Az új nézet.

4.14 src/game/leaderboard.c fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <wchar.h>
#include "utils.h"
#include "app.h"
#include "game/player.h"
```

Függvények

- Leaderboard * Leaderboard__construct (char const *records_file_path)

  Létrehozza a dicsőséglistát.
- void Leaderboard__destroy (Leaderboard *this)

Felszabadítja a megadott `Leaderboard` struktúra által lefoglalt memóriát.

- bool Leaderboard__load (Leaderboard *this)

  Betölti a rekordfájl tartalmát.

- void Leaderboard__sort (Leaderboard *this)

  A beolvasott rekordok csökkenő sorba rendezése.

- bool Leaderboard__insert_new (Leaderboard *this, Player *new)

  Új rekord beszúrása és mentése.

4.14.1 Függvények dokumentációja

4.14.1.1 Leaderboard__construct()

```
Leaderboard * Leaderboard__construct (
            char const * records_file_path )
```

Létrehozza a dicsőséglistát.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Leaderboard__destroy` függvényt.

Paraméterek

records_file_path	A fájl elérési útvonala, ahonnan beolvassa a játékosneveket és rekordokat.

Visszatérési érték

Pointer az újonnan létrehozott `Leaderboard`-ra.

4.14.1.2 Leaderboard__destroy()

```
void Leaderboard__destroy (
            Leaderboard * this )
```

Felszabadítja a megadott `Leaderboard` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Leaderboard` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.14.1.3 Leaderboard__insert_new()

```
bool Leaderboard__insert_new (
            Leaderboard * this,
            Player * new )
```

Új rekord beszúrása és mentése.

Megjegyzés: A visszatérési érték nem feltétlen van kihasználva, elhanyagolható.

Paraméterek

this	A `Leaderboard` struktúra, amihez tartozó rekordfájlba történjen a mentés.
new	A potenciálisan új, rekordot elérő játékos.

Visszatérési érték

Már tartalmazta e a dicsőséglista a játékost.

4.14.1.4 Leaderboard__load()

```
bool Leaderboard__load (
            Leaderboard * this )
```

Betölti a rekordfájl tartalmát.

Érdemes meghívni a `Leaderboard__sort`-ot ezután.

Paraméterek

this	A `Leaderboard` struktúra, amelyhez már tartozik egy rekordfájl.

Visszatérési érték

Hibamentesen futott e le.

4.14.1.5 Leaderboard__sort()

```
void Leaderboard__sort (
            Leaderboard * this )
```

A beolvasott rekordok csökkenő sorba rendezése.

Paraméterek

this	A `Leaderboard` struktúra, amely a beolvasott rekordokat tárolja.

4.15 src/game/players.c fájlreferencia

```
#include <stdio.h>
#include <string.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "utils.h"
#include "app.h"
#include "game/player.h"
#include "game/meeple.h"
```

Függvények

- Meeple Meeple__construct (SDL_Texture *texture)

  Létrehoz egy alattvalót.
- Player Player__construct (SDL_Renderer *renderer, TTF_Font *font, char *name, char const *meeple_↩ outfit)

  Létrehoz egy játékost.
- void Player__destroy (Player *this)

  Felszabadítja a megadott `Player` struktúra által lefoglalt memóriát.
- void Player__render (Player *this, SDL_Renderer *renderer, TTF_Font *font)

  A megadott játékos statisztájának lerenderelése.
- void Player__place_meeple (Player *this, SDL_Point tile_index)

  Alattvaló lehelyezése.
- void Player__reclaim_meeple (Player *this, Meeple *to_reclaim)

  Lehelyezett alattvaló visszaszerzése.
- void Player__add_to_score (Player *this, unsigned int add)

  Játékos pontszámának növelése.

4.15.1 Függvények dokumentációja

4.15.1.1 Meeple__construct()

```
Meeple Meeple__construct (
            SDL_Texture * texture )
```

Létrehoz egy alattvalót.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Meeple__destroy` függvényt.

Paraméterek

texture	A kinézete.

Visszatérési érték

Az új `Meeple` struktúra.

4.15.1.2 Player__add_to_score()

```
void Player__add_to_score (
            Player * this,
            unsigned int add )
```

Játékos pontszámának növelése.

Beállítja az `update_score`-t, a következő render esetén frissíti a számlálót.

Paraméterek

this	A `Player`, amelynek a pontszámát növeljük.
add	A hozzáadott mennyiség.

4.15.1.3 Player__construct()

```
Player Player__construct (
            SDL_Renderer * renderer,
            TTF_Font * font,
            char * name,
            char const * meeple_outfit )
```

Létrehoz egy játékost.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Player__destroy` függvényt.

Paraméterek

renderer	A renderer amivel a főablakra rajzolunk.
font	A statisztikák kiírásához használt betűtípus.
name	A neve.
meeple_outfit	Az alattvalóinak kinézete (képfájl).

Visszatérési érték

Az új Player.

4.15.1.4 Player__destroy()

```
void Player__destroy (
            Player * this )
```

Felszabadítja a megadott Player struktúra által lefoglalt memóriát.

Paraméterek

this	A Player struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.15.1.5 Player__place_meeple()

```
void Player__place_meeple (
            Player * this,
            SDL_Point tile_index )
```

Alattvaló lehelyezése.

Nem viszgálja meg, hogy a mezőkártya valóban megfelelő e.

Paraméterek

this	Az a Player, amelynek egy alattvalóját le kell helyezni.
tile_index	A mezőkártya koordinátája a játéktáblán, amire az alattvaló le lesz téve.

4.15.1.6 Player__reclaim_meeple()

```
void Player__reclaim_meeple (
            Player * this,
            Meeple * to_reclaim )
```

Lehelyezett alattvaló visszaszerzése.

Paraméterek

this	A Player struktúra, amelynek egy alattvalóját vissza kell szerezni.
to_reclaim	A visszaszerzendő Meeple.

4.15.1.7 Player__render()

```
void Player__render (
            Player * this,
            SDL_Renderer * renderer,
            TTF_Font * font )
```

A megadott játékos statisztájának lerenderelése.

Paraméterek

this	A `Player` struktúra, aminek a statisztikáját akarjuk megjeleníteni.
renderer	A renderer amivel a főablakra rajzolunk.
font	A statisztikák kiírásához használt betűtípus.

4.16 src/game/tile.c fájlreferencia

```
#include <math.h>
#include <SDL2/SDL.h>
#include "app.h"
#include "game/tile.h"
```

Függvények

- CardPile * CardPile__construct (void)

  Létrehozza a kártyapaklit.
- void CardPile__destroy (CardPile *this)

  Felszabadítja a megadott `CardPile` strúktúrát (a legutolsó elemen kell meghívni).
- CardPile * CardPile__pop (CardPile *this, TileType *popped)

  Leveszi és eltárolja a pakli legfelső elemét.
- CardPile * CardPile__push (CardPile *this, TileType new_type)

  A megadott paklira rátesz egy új kártyát.
- void Tile__construct (Tile *this, TileType type, SDL_Point board_coords, SDL_FPoint offset)

  Létrehoz egy mezőkártyát.
- bool Tile__point_in_tile (Tile *this, SDL_FPoint pt)

  Ellenőrzi hogy egy pont az adott kártya területében van e.
- void Tile__move_by (Tile *this, float mvx, float mvy)

  Elmozgatja a megadott mezőkártyát egy adott irányba.
- void Tile__rotate (Tile *this)

  Órajárásával megegyező irányába elforgatja 90 fokkal a megadott mezőkártyát.
- void Tile__set_rotation (Tile *this, unsigned short new_rotation)

  A megadott mezőkártyának elforgatását beállítja pontosan a kapott értékre.
- void Tile__set_type (Tile *this, TileType new_type, unsigned short new_rotation)

  Adott mezőkártya típusának megváltoztatása.

4.16.1 Függvények dokumentációja

4.16.1.1 CardPile__construct()

```
CardPile * CardPile__construct (
            void  )
```

Létrehozza a kártyapaklit.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `CardPile__destroy` függvényt.

Visszatérési érték

Pointer az új `CardPile` legelső elemére (mindig EMPTY típust tárol) vagy NULL ha hiba volt.

4.16.1.2 CardPile__destroy()

```
void CardPile__destroy (
            CardPile * this )
```

Felszabadítja a megadott `CardPile` strúktúrát (a legutolsó elemen kell meghívni).

Rekúrzívan működik.

Paraméterek

this	A `CardPile`, aminek a lefoglalt memóriáját fel kell szabadítani.

4.16.1.3 CardPile__pop()

```
CardPile * CardPile__pop (
            CardPile * this,
            TileType * popped )
```

Leveszi és eltárolja a pakli legfelső elemét.

Paraméterek

this	A pakli aminek leveszi a legfelső elemét.
popped	Ahova eltárolja a kapott kártyatípust.

Visszatérési érték

Pointer a `CardPile` új legfelső elemére vagy NULL ha hiba volt.

4.16.1.4 CardPile__push()

```
CardPile * CardPile__push (
            CardPile * this,
            TileType new_type )
```

A megadott paklira rátesz egy új kártyát.

Paraméterek

this	A pakli amire rá kell tenni egy új kártyát.
new_type	A tárolandó kártyatípus.

Visszatérési érték

Pointer az új legfelső elemre vagy NULL ha hiba volt.

4.16.1.5 Tile__construct()

```
void Tile__construct (
            Tile * this,
            TileType type,
            SDL_Point board_coords,
            SDL_FPoint offset )
```

Létrehoz egy mezőkártyát.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Tile__destroy` függvényt.

Paraméterek

this	A létrejövő kártya.
type	A kártya típusa.
board_coords	A cella koordinátái a táblán (0 és BOARD_SIZE közt).
offset	A tábla koordinátái.

4.16.1.6 Tile__move_by()

```
void Tile__move_by (
            Tile * this,
```

```
            float mvx,
            float mvy )
```

Elmozgatja a megadott mezőkártyát egy adott irányba.

Paraméterek

this	Az adott `Tile`.
mvx	Mozgatás az x tengelyen.
mvy	Mozgatás az y tengelyen.

4.16.1.7 Tile__point_in_tile()

```
bool Tile__point_in_tile (
            Tile * this,
            SDL_FPoint pt )
```

Ellenőrzi hogy egy pont az adott kártya területében van e.

Paraméterek

this	A vizsgálandó mezőkártya.
pt	A vizsgálandó pont.

Visszatérési érték

Benne van e a pont.

4.16.1.8 Tile__rotate()

```
void Tile__rotate (
            Tile * this )
```

Órajárásával megegyező irányába elforgatja 90 fokkal a megadott mezőkártyát.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.

4.16.1.9 Tile__set_rotation()

```
void Tile__set_rotation (
            Tile * this,
            unsigned short new_rotation )
```

A megadott mezőkártyának elforgatását beállítja pontosan a kapott értékre.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.
new_rotation	Az elforgatás értéke.

4.16.1.10 Tile__set_type()

```
void Tile__set_type (
            Tile * this,
            TileType new_type,
            unsigned short new_rotation )
```

Adott mezőkártya típusának megváltoztatása.

Az oldalak `ConnectionType`-jait is megfelelően beállítja.

Paraméterek

this	Az adott `Tile`.
new_type	Az új típusa.
new_rotation	-

4.17 src/main.c fájlreferencia

```
#include <SDL2/SDL_log.h>
#include "app.h"
```

Függvények

- int main (void)

  A program entry pointja.

4.17.1 Függvények dokumentációja

4.17.1.1 main()

```
int main (
            void  )
```

A program entry pontja.

Létrehozza és inicializálja a programciklust, illetve biztosítja a lefoglalt memória helyes felszabadulását.

Visszatérési érték

0

4.18 src/textures.c fájlreferencia

```
#include <SDL2/SDL.h>
#include "utils.h"
#include "game/tile.h"
```

Függvények

- TilesetWrapper TilesetWrapper__construct (SDL_Renderer *renderer)

  Létrehoz egy új `TilesetWrapper`-t.
- void TilesetWrapper__destroy (TilesetWrapper *this)

  Felszabadítja a megadott `TilesetWrapper` struktúrában tárolt textúra atlaszt.
- SDL_Rect get_texture_rect_for (TileType type)

  A megadott kártyatípushoz tartozó textúra pozícióját számolja ki az atlaszon.

4.18.1 Függvények dokumentációja

4.18.1.1 get_texture_rect_for()

```
SDL_Rect get_texture_rect_for (
            TileType type )
```

A megadott kártyatípushoz tartozó textúra pozícióját számolja ki az atlaszon.

Paraméterek

type	A mezőkártya típusa, amelynek a textúráját szeretnénk.

Visszatérési érték

A textúra pozíciója az atlaszon.

4.18.1.2 TilesetWrapper__construct()

```
TilesetWrapper TilesetWrapper__construct (
            SDL_Renderer * renderer )
```

Létrehoz egy új `TilesetWrapper`-t.

Singletonként kell használni a programban. Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `TilesetWrapper__destroy` függvényt.

Paraméterek

renderer	A renderer amivel a főablakra rajzolunk.

Visszatérési érték

Az új atlasz wrapper.

4.18.1.3 TilesetWrapper__destroy()

```
void TilesetWrapper__destroy (
            TilesetWrapper * this )
```

Felszabadítja a megadott `TilesetWrapper` struktúrában tárolt textúra atlaszt.

Paraméterek

this	A `TilesetWrapper`, aminek a lefoglalt memóriáját fel kell szabadítani.

4.19 src/ui/game.c fájlreferencia

```
#include <stdlib.h>
#include "game/tile.h"
#include "utils.h"
#include "ui.h"
#include "app.h"
```

Függvények

- void Carcassone__Game__construct (Carcassone *this)

  Létrehozza a játéknézetet.
- void Carcassone__Game__destroy (Carcassone *this)

  Felszabadítja a megadott `Carcassone` struktúrához tartozó `GameScreen` által lefoglalt memóriát.
- void Carcassone__Game__handle_input (Carcassone *this, float dt)

  Inputok kezelése játéknézetben.
- void Carcassone__Game__render (Carcassone *this)

  Játéknézet megjelenítése.
- bool Carcassone__Game__check_names_valid (Carcassone *this)

  Leellenőrzi, hogy a két megadott játékos név helyes e.
- void Carcassone__Game__init_players (Carcassone *this)

  Játékosok létrehozása.
- void Carcassone__Game__init_pile (Carcassone *this)

  Kártyapakli létrehozása.
- void Carcassone__Game__init_board (Carcassone *this)

  Játéktábla létrehozása.
- void Carcassone__Game__render_board (Carcassone *this)

  Játéktábla renderelése.
- void Carcassone__Game__render_drawn_tile (Carcassone *this)

  A pakli tetején levő kártya és a pakli méretének renderelése.
- void Carcassone__Game__render_meeples (Carcassone *this)

  A lehelyezett alattvalók renderelése.
- void Carcassone__Game__render_game_over (Carcassone *this)

  A "game over" állapot megjelenítése.
- void Carcassone__Game__indicate_possible_placements (Carcassone *this)
- bool Carcassone__Game__check_if_possible (Carcassone *this)

  Megnézi, hogy a húzott kártyát valahova le lehet e tenni.
- void Carcassone__Game__calculate_points (Carcassone *this)

  A megfelelő pontszámító függvényeknek delegálja a feladatot.
- void Carcassone__Game__calculate_scores_for_cloister (Carcassone *this, Tile *tile)

  Kolostorok pontszámítása.
- void Carcassone__Game__calculate_scores_for_graph (Carcassone *this, Tile *tile, ConnectionType conn_type)

  Utak és várak pontszámítása.
- void Carcassone__Game__render_player_stats (Carcassone *this)

  A splash cím renderelése.
- void Carcassone__Game__move_board (Carcassone *this, float dt)

  Játéktábla mozgatása.
- bool Carcassone__Game__check_surrounding_tiles (Carcassone *this, SDL_Point tcoords)

  Helyes pozíció ellenőrzése.
- bool Carcassone__Game__draw_new (Carcassone *this)

  Új kártya húzása a pakli tetejéről.
- void Carcassone__Game__wrapup (Carcassone *this, bool concede)

  "Game over" szituáció menedzselése.

4.19.1 Függvények dokumentációja

4.19.1.1 Carcassone__Game__calculate_points()

```
void Carcassone__Game__calculate_points (
            Carcassone * this )
```

A megfelelő pontszámító függvényeknek delegálja a feladatot.

Paraméterek

this	

4.19.1.2 Carcassone__Game__calculate_scores_for_cloister()

```
void Carcassone__Game__calculate_scores_for_cloister (
            Carcassone * this,
            Tile * tile )
```

Kolostorok pontszámítása.

Paraméterek

this	
tile	

4.19.1.3 Carcassone__Game__calculate_scores_for_graph()

```
void Carcassone__Game__calculate_scores_for_graph (
            Carcassone * this,
            Tile * tile,
            ConnectionType conn_type )
```

Utak és várak pontszámítása.

Paraméterek

this	
tile	
conn_type	

4.19.1.4 Carcassone__Game__check_if_possible()

```
bool Carcassone__Game__check_if_possible (
            Carcassone * this )
```

Megnézi, hogy a húzott kártyát valahova le lehet e tenni.

Paraméterek

this	

Visszatérési érték

Le lehet e tenni a húzott kártyát valahova.

4.19.1.5 Carcassone__Game__check_names_valid()

```
bool Carcassone__Game__check_names_valid (
            Carcassone * this )
```

Leellenőrzi, hogy a két megadott játékos név helyes e.

Ellenőrzi a hosszúságukat és hogy nem e ugyanazok.

Paraméterek

this	A `Carcassone` struktúra, amihez a nézet tartozik.

Visszatérési érték

Helyesek e.

4.19.1.6 Carcassone__Game__check_surrounding_tiles()

```
bool Carcassone__Game__check_surrounding_tiles (
            Carcassone * this,
            SDL_Point tcoords )
```

Helyes pozíció ellenőrzése.

Paraméterek

this	A `Carcassone struktúra`, ami tartalmazza a játéknézetet.
tcoords	A kapott kártya potenciális helye a táblán.

Visszatérési érték

Letehető-e a megfelelő pozícióba az adott kártya.

4.19.1.7 Carcassone__Game__construct()

```
void Carcassone__Game__construct (
            Carcassone * this )
```

Létrehozza a játéknézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Game__destroy` függvényt.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a játéknézetét.

4.19.1.8 Carcassone__Game__destroy()

```
void Carcassone__Game__destroy (
            Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúrához tartozó `GameScreen` által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.19.1.9 Carcassone__Game__draw_new()

```
bool Carcassone__Game__draw_new (
            Carcassone * this )
```

Új kártya húzása a pakli tetejéről.

Paraméterek

this	A `Carcassone` struktúra.

Visszatérési érték

Lehet e még húzni (ha üres a pakli: false).

4.19.1.10 Carcassone__Game__handle_input()

```
void Carcassone__Game__handle_input (
            Carcassone * this,
            float dt )
```

Inputok kezelése játéknézetben.

Paraméterek

this	A Carcassone struktúra, amihez tartozik a játéknézet.
dt	Deltaidő.

4.19.1.11 Carcassone__Game__indicate_possible_placements()

```
void Carcassone__Game__indicate_possible_placements (
            Carcassone * this )
```

4.19.1.12 Carcassone__Game__init_board()

```
void Carcassone__Game__init_board (
            Carcassone * this )
```

Játéktábla létrehozása.

Létrehozza a játéktáblát és le is helyezi a kezdőkártyát.

Paraméterek

this	A Carcassone struktúra.

4.19.1.13 Carcassone__Game__init_pile()

```
void Carcassone__Game__init_pile (
            Carcassone * this )
```

Kártyapakli létrehozása.

Véletlenszerűen megkeveri a paklit és létrehozza a mezőkártyákat mindegyikhez.

Paraméterek

this	A Carcassone struktúra.

4.19.1.14 Carcassone__Game__init_players()

```
void Carcassone__Game__init_players (
            Carcassone * this )
```

Játékosok létrehozása.

Inicializálja a két játékost előre megadott adatok alapján (ezek a GameScreen-ben találhatók).

Paraméterek

this	A Carcassone struktúra.

4.19.1.15 Carcassone__Game__move_board()

```
void Carcassone__Game__move_board (
            Carcassone * this,
            float dt )
```

Játéktábla mozgatása.

A nyilak segítségével a játéktábla látható részét mozgatja.

Paraméterek

this	A Carcassone struktúra, ami tartalmazza az SDL kontextust.
dt	Deltaidő.

4.19.1.16 Carcassone__Game__render()

```
void Carcassone__Game__render (
            Carcassone * this )
```

Játéknézet megjelenítése.

Paraméterek

this	A Carcassone struktúra, amihez a nézet tartozik.

4.19.1.17 Carcassone__Game__render_board()

```
void Carcassone__Game__render_board (
            Carcassone * this )
```

Játéktábla renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.

4.19.1.18 Carcassone__Game__render_drawn_tile()

```
void Carcassone__Game__render_drawn_tile (
            Carcassone * this )
```

A pakli tetején levő kártya és a pakli méretének renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.

4.19.1.19 Carcassone__Game__render_game_over()

```
void Carcassone__Game__render_game_over (
            Carcassone * this )
```

A "game over" állapot megjelenítése.

Megjeleníti a koronát a nyertes játékos stat panelje felett.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.

4.19.1.20 Carcassone__Game__render_meeples()

```
void Carcassone__Game__render_meeples (
            Carcassone * this )
```

A lehelyezett alattvalók renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza a renderert.

4.19.1.21 Carcassone__Game__render_player_stats()

```
void Carcassone__Game__render_player_stats (
            Carcassone * this )
```

A splash cím renderelése.

Paraméterek

this	A `Carcassone` struktúra, ami tartalmazza az SDL kontextust.

4.19.1.22 Carcassone__Game__wrapup()

```
void Carcassone__Game__wrapup (
            Carcassone * this,
            bool concede )
```

"Game over" szituáció menedzselése.

Nyertes kiválasztása, dicsőséglista frissítése és visszatérés a menübe.

Paraméterek

this	A `Carcassone` struktúra, amihez a a nézet tartozik.
concede	Feladás történt e.

4.20 src/ui/lboard.c fájlreferencia

```
#include <stdlib.h>
#include "utils.h"
#include "ui.h"
#include "app.h"
```

Függvények

- void Carcassone__Lboard__construct (Carcassone *this)

Létrehozza a dicsőséglistanézetet.

- void Carcassone__Lboard__destroy (Carcassone *this)

  Felszabadítja a megadott `Carcassone` struktúrához tartozó `LeaderboardScreen` által lefoglalt memóriát.

- void Carcassone__Lboard__handle_input (Carcassone *this)

  Inputok kezelése dicsőglistanézetben.

- void Carcassone__Lboard__init_list_texture (Carcassone *this)

  Létrehozza a dicsőséglistanézethez a rekordokat.

- void Carcassone__Lboard__render (Carcassone *this)

  Dicsőséglistanézet megjelenítése.

4.20.1 Függvények dokumentációja

4.20.1.1 Carcassone__Lboard__construct()

```
void Carcassone__Lboard__construct (
            Carcassone * this )
```

Létrehozza a dicsőséglistanézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Lboard__destroy` függvényt.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a dicsőséglistanézetét.

4.20.1.2 Carcassone__Lboard__destroy()

```
void Carcassone__Lboard__destroy (
            Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúrához tartozó `LeaderboardScreen` által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.20.1.3 Carcassone__Lboard__handle_input()

```
void Carcassone__Lboard__handle_input (
            Carcassone * this )
```

Inputok kezelése dicsőglistanézetben.

Paraméterek

this	A `Carcassone` struktúra, amihez tartozik a dicsőglistanézet.

4.20.1.4 Carcassone__Lboard__init_list_texture()

```
void Carcassone__Lboard__init_list_texture (
            Carcassone * this )
```

Létrehozza a dicsőséglistanézethez a rekordokat.

Akkor kell meghívni, ha frissül a rekordfájl.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a dicsőséglistanézetét.

4.20.1.5 Carcassone__Lboard__render()

```
void Carcassone__Lboard__render (
            Carcassone * this )
```

Dicsőséglistanézet megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amihez a dicsőséglista tartozik.

4.21 src/ui/menu.c fájlreferencia

```
#include <stdlib.h>
#include "utils.h"
#include "ui.h"
#include "app.h"
```

Függvények

- void Carcassone__Menu__construct (Carcassone *this)

  Létrehozza a menünézetet.
- void Carcassone__Menu__destroy (Carcassone *this)

Felszabadítja a megadott `Carcassone` struktúrához tartozó `MenuScreen` által lefoglalt memóriát.

- void Carcassone__Menu__handle_input (Carcassone *this)

  Inputok kezelése menünézetben.

- void Carcassone__Menu__render (Carcassone *this)

  Menünézet megjelenítése.

4.21.1 Függvények dokumentációja

4.21.1.1 Carcassone__Menu__construct()

```
void Carcassone__Menu__construct (
            Carcassone * this )
```

Létrehozza a menünézetet.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Menu__destroy` függvényt.

Paraméterek

this	A `Carcassone` struktúra, amelynek létrehozza a menünézetét.

4.21.1.2 Carcassone__Menu__destroy()

```
void Carcassone__Menu__destroy (
            Carcassone * this )
```

Felszabadítja a megadott `Carcassone` struktúrához tartozó `MenuScreen` által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.21.1.3 Carcassone__Menu__handle_input()

```
void Carcassone__Menu__handle_input (
            Carcassone * this )
```

Inputok kezelése menünézetben.

Paraméterek

this	A `Carcassone` struktúra, amihez tartozik a menünézet.

4.21.1.4 Carcassone__Menu__render()

```
void Carcassone__Menu__render (
            Carcassone * this )
```

Menünézet megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amihez a menü tartozik.

4.22 src/ui/ui.c fájlreferencia

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_ttf.h>
#include "ui.h"
#include "app.h"
#include "utils.h"
```

Függvények

- Button Carcassone__Button__construct (Carcassone *this, TTF_Font *font, char *label, SDL_Rect global↩ _rect, SDL_Color bg_color, SDL_Color text_color, bool center)

  Létrehoz egy gombot.
- void Carcassone__Button__destroy (Carcassone *this, Button *button)

  Felszabadítja a megadott `Button` struktúra által lefoglalt memóriát.
- void Carcassone__Button__render (Carcassone *this, Button *button)

  A gomb megjelenítése.
- Prompt Carcassone__Prompt__construct (Carcassone *this, TTF_Font *font, char *default_label, SDL_Rect global_rect, SDL_Color bg_color, SDL_Color text_color)

  Létrehoz egy szöveginputot.
- void Carcassone__Prompt__destroy (Carcassone *this, Prompt *prompt)

  Felszabadítja a megadott `Prompt` struktúra által lefoglalt memóriát.
- void Carcassone__Prompt__render (Carcassone *this, Prompt *prompt)

  A szöveginput megjelenítése.
- void Carcassone__Prompt__edit (Carcassone *this, Prompt *prompt, char *new_label, bool concat)

  Vagy hozzátold a szöveginput címkéjéhez, vagy lecseréli azt.

4.22.1 Függvények dokumentációja

4.22.1.1 Carcassone__Button__construct()

```
Button Carcassone__Button__construct (
            Carcassone * this,
            TTF_Font * font,
            char * label,
            SDL_Rect global_rect,
            SDL_Color bg_color,
            SDL_Color text_color,
            bool center )
```

Létrehoz egy gombot.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a `Carcassone__Button__destroy` függvényt.

Paraméterek

this	A renderert tartalmazó `Carcassone` struktúra.
font	A címkéhez használt betűtípus.
label	A címke tartalma.
global_rect	A gomb globális pozíciója (az ablakon).
bg_color	A gomb színe.
text_color	A címke színe.
center	A container-ben középre legyen e igazítva a címke.

Visszatérési érték

Az új `Button`.

4.22.1.2 Carcassone__Button__destroy()

```
void Carcassone__Button__destroy (
            Carcassone * this,
            Button * button )
```

Felszabadítja a megadott `Button` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek az ablakára az adott gomb renderelve van (nem használt, régi verzió miatt).
button	A `Button` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.22.1.3 Carcassone__Button__render()

```
void Carcassone__Button__render (
            Carcassone * this,
            Button * button )
```

A gomb megjelenítése.

Paraméterek

this	A Carcassone struktúra, amelyhez a renderer tartozik.
button	A megjelenítendő gomb.

4.22.1.4 Carcassone__Prompt__construct()

```
Prompt Carcassone__Prompt__construct (
            Carcassone * this,
            TTF_Font * font,
            char * default_label,
            SDL_Rect global_rect,
            SDL_Color bg_color,
            SDL_Color text_color )
```

Létrehoz egy szöveginputot.

Megjegyzés: A lefoglalt memória megfelelő felszabadításához meg kell hívni a Carcassone__Prompt__↩ destroy függvényt.

Paraméterek

this	A renderert tartalmazó Carcassone struktúra.
font	A címkéhez használt betűtípus.
default_label	A címke tartalma.
global_rect	A gomb globális pozíciója (az ablakon).
bg_color	A gomb színe.
text_color	A címke színe.

Visszatérési érték

Az új Prompt.

4.22.1.5 Carcassone__Prompt__destroy()

```
void Carcassone__Prompt__destroy (
            Carcassone * this,
            Prompt * prompt )
```

Felszabadítja a megadott `Prompt` struktúra által lefoglalt memóriát.

Paraméterek

this	A `Carcassone` struktúra, aminek az ablakára az adott szöveginput renderelve van (nem használt, régi verzió miatt).
prompt	A `Prompt` struktúra, aminek a lefoglalt memóriáját fel kell szabadítani.

4.22.1.6 Carcassone__Prompt__edit()

```
void Carcassone__Prompt__edit (
            Carcassone * this,
            Prompt * prompt,
            char * new_label,
            bool concat )
```

Vagy hozzátold a szöveginput címkéjéhez, vagy lecseréli azt.

Paraméterek

this	A renderert tartalmazó `Carcassone` struktúra.
prompt	A szöveginput.
new_label	Az új címke vagy a hozzátoldott szöveg.
concat	Hozzátoldás e.

4.22.1.7 Carcassone__Prompt__render()

```
void Carcassone__Prompt__render (
            Carcassone * this,
            Prompt * prompt )
```

A szöveginput megjelenítése.

Paraméterek

this	A `Carcassone` struktúra, amelyhez a renderer tartozik.
prompt	A megjelenítendő szöveginput.

4.23 src/utils.c fájlreferencia

```
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <wchar.h>
#include <SDL2/SDL_render.h>
#include <SDL2/SDL_surface.h>
#include "utils.h"
```

Függvények

- size_t mb_strlen (char const *str)

  Multibyte (akár UTF8) karaktereket tartalmazó sztring "látható karaktereinek" számának meghatározása.
- char * strdup_ (char const *str)

  `strdup` *POSIX függvényhez hasonló.*
- char * strcatdyn (char *original, char const *to_cat)

  Dinamikus sztringhez hozzátoldás.
- char * mb_remove_last_char_dyn (char *str)

  Dinamikus sztring utolsó karakterének (multibyte) eltávolítása.
- void destroy_SDL_Texture (SDL_Texture *texture)

  `SDL_Texture` *felszabadítása, de kezeli azt is ha NULL a megadott textúra.*
- SDL_Texture * create_SDL_texture_from_BMP (SDL_Renderer *renderer, char const *source_path)

  `SDL_Texture` *létrehozása egy BMP képfájlból.*

4.23.1 Függvények dokumentációja

4.23.1.1 create_SDL_texture_from_BMP()

```
SDL_Texture * create_SDL_texture_from_BMP (
            SDL_Renderer * renderer,
            char const * source_path )
```

`SDL_Texture` létrehozása egy BMP képfájlból.

Megjegyzés: A létrejött textúra a `destroy_SDL_Texture`-rel törlendő.

Paraméterek

renderer	A főablakhoz tartozó renderer.
source_path	A BMP képfájl elérési útvonala.

Visszatérési érték

A létrehozott textúrára mutató pointer.

4.23.1.2 destroy_SDL_Texture()

```
void destroy_SDL_Texture (
            SDL_Texture * texture )
```

`SDL_Texture` felszabadítása, de kezeli azt is ha NULL a megadott textúra.

Paraméterek

texture	A törlendő textúra.

4.23.1.3 mb_remove_last_char_dyn()

```
char * mb_remove_last_char_dyn (
            char * str )
```

Dinamikus sztring utolsó karakterének (multibyte) eltávolítása.

Megjegyzés: Az eredeti sztringet felszabadítja.

Paraméterek

str	A dinamikusan foglalt sztring.

Visszatérési érték

Egy új dinamikusan foglalt sztring, amelynek tartalma az `str`-ével megegyezik, kivéve annak utolsó karaktere.

4.23.1.4 mb_strlen()

```
size_t mb_strlen (
            char const * str )
```

Multibyte (akár UTF8) karaktereket tartalmazó sztring "látható karaktereinek" számának meghatározása.

Paraméterek

str	A sztring aminek meghatározzuk a hosszát.

Visszatérési érték

A megadott sztring hossza.

4.23.1.5 strcatdyn()

```
char * strcatdyn (
            char * original,
            char const * to_cat )
```

Dinamikus sztringhez hozzátoldás.

Megjegyzés: Az eredeti sztringet felszabadítja.

Paraméterek

original	A dinamikusan foglalt sztring, amelyhez hozzátoldunk.
to_cat	A hozzátoldandó sztring (nem feltétlen dinamikus).

Visszatérési érték

Egy új dinamikusan foglalt sztring, amelynek tartalma az `original` és `to_cat`-é.

4.23.1.6 strdup_()

```
char * strdup_ (
            char const * str )
```

`strdup` POSIX függvényhez hasonló.

Paraméterek

str	A sztring amit dinamikusan le kell foglalni.

Visszatérési érték

Az új, dinamikusan foglalt sztring.