Univerzitet u Sarajevu
Elektrotehnički fakultet Sarajevo
Odsjek za računarstvo i informatiku

DOKUMENTACIJA IMPLEMENTACIJE

Gomoku (5-in-a-row)

Ahmedin Hasanović, 18646

Ema Kalmar, 18684

Implementacija sistema

Funkcije korištene u sistemu

```
bool singlePressed(int x, int y) {
    return x >= 75 && x <= 165 && y >= 80 && y <= 105;
}

bool multiPressed(int x, int y) {
    return x >= 75 && x <= 165 && y >= 140 && y <= 165;
}

bool continuePressed(int x, int y) {
    return x >= 65 && x <= 170 && y >= 80 && y <= 110;
}

bool restartPressed(int x, int y) {
    return x >= 65 && x <= 170 && y >= 130 && y <= 160;
}

bool homePressed(int x, int y) {
    return x >= 65 && x <= 170 && y >= 180 && y <= 210;
}

bool pausePressed(int x, int y) {
    return x >= 215 && x <= 227 && y >= 15 && y <= 25;
}
```

Ove funkcije su veoma jednostavne i one pomažu pri detektovanju da li je kliknuto određeno dugme (1 PLAYER ili 2 PLAYER na početnom zaslonu, pauza tijekom igre te sve opcije moguće na pauzi (continue, restart, home) kao i mogućnost odlaska na početni zaslon nakon završetka igre.

```
void singlePlayerView() {
    // Glavna pozadina
    BSP_LCD_SetTextColor(LCD_COLOR_BROWN);
    BSP_LCD_FillRect(0,0,240,240);

    // Horizontalne rešetke za polja
    BSP_LCD_SetTextColor(LCD_COLOR_BLACK);
    for (int i = 45; i <= 210; i += 15) BSP_LCD_DrawLine(0,i,240,i);

    // Vertikalne rešetke
    for (int i = 15; i <= 225; i += 15) BSP_LCD_DrawLine(i,45,i,210);

    // Natpis iznad rešetki
    BSP_LCD_SetFont(&Font16);
    BSP_LCD_SetTextColor(LCD_COLOR_YELLOW);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(20,15, (uint8_t *)"CLICK ON FIELD", LEFT_MODE);

    // Player1 tekst
    BSP_LCD_SetFont(&Font12);
    BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(5,220,(uint8_t *)"Player1", LEFT_MODE);

    // BOT tekst
    BSP_LCD_SetFont(&Font12);
    BSP_LCD_SetTextColor(LCD_COLOR_BLACK);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(200,220,(uint8_t *)"BOT", LEFT_MODE);

    // Dugme za pauzu
    BSP_LCD_SetTextColor(LCD_COLOR_YELLOW);
    BSP_LCD_FillRect(215, 15, 5, 10);
    BSP_LCD_FillRect(222, 15, 5, 10);

    if (last == 4) player = State::WHITE;
    if (player == State::WHITE) pointingToPlayer(1);
    else pointingToPlayer(2);
}
```

```
void multiPlayerView() {
    // Glavna pozadina
    BSP_LCD_SetTextColor(LCD_COLOR_BROWN);
    BSP_LCD_FillRect(0,0,240,240);

    // Horizontalne rešetke za polja
    BSP_LCD_SetTextColor(LCD_COLOR_BLACK);
    for (int i = 45; i <= 210; i += 15) BSP_LCD_DrawLine(0,i,240,i);

    // Vertikalne rešetke
    for (int i = 15; i <= 225; i += 15) BSP_LCD_DrawLine(i,45,i,210);

    // Natpis iznad rešetki
    BSP_LCD_SetFont(&Font16);
    BSP_LCD_SetTextColor(LCD_COLOR_YELLOW);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(20,15, (uint8_t *)"CLICK ON FIELD", LEFT_MODE);

    // Player1 tekst
    BSP_LCD_SetFont(&Font12);
    BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(5,220,(uint8_t *)"Player1", LEFT_MODE);

    // Player2 tekst
    BSP_LCD_SetFont(&Font12);
    BSP_LCD_SetTextColor(LCD_COLOR_BLACK);
    BSP_LCD_SetBackColor(LCD_COLOR_BROWN);
    BSP_LCD_DisplayStringAt(185,220,(uint8_t *)"Player2", LEFT_MODE);

    // Dugme za pauzu
    BSP_LCD_SetTextColor(LCD_COLOR_YELLOW);
    BSP_LCD_FillRect(215, 15, 5, 10);
    BSP_LCD_FillRect(222, 15, 5, 10);

    if (last == 4) player = State::WHITE;
    if (player == State::WHITE) pointingToPlayer(1);
    else pointingToPlayer(2);
}
```

Funkcije *singlePlayerView()* i *multiPlayerView()* koriste se za prikaz polja na kojem se igra i one se jedino razlikuju u tome što prva ispisuje „BOT“ (jer se igra protiv računara), a druga ispisuje „PLAYER 2“ (jer se igra protiv druge osobe).

```
void pointingToPlayer(int player) {
    // Okviri koji pokazuju koji igrač je na redu
    if (player == 1) BSP_LCD_DrawRect(3,218,52,14);
    else if (player == 2) BSP_LCD_DrawRect(183,218,52,14);
}
```

Funkcija koja pomjera žuti pravougaonik lijevo ili desno, u zavisnosti koji je igrač na potezu.

```
class Board {
    MatrixField fields;
public:
    Board(){}
    Board (MatrixField f) { fields = f; }
    void makeBoard() {⟺}
    bool checkAvailability(int x, int y) {⟺}
    bool endOfGame() {⟺}
    bool draw() {⟺}
    void drawView() {⟺}
    void winnerView (int p) {⟺}
    void update() {⟺}
    void insert (int x, int y) {⟺}
    void bot() {⟺}
    void size() {⟺}
};
```

Tu je i glavna klasa *Board* koja predstavlja matricu u koju smještamo crne i bijele figure. Neke od metoda su:

- *checkAvailability* – provjerava da li smo kliknuli na već zauzeto mjesto
- *endOfGame* – da li smo završili igru
- *draw* – da li su popunjena sva polja bez pobjednika
- *drawView* – izgled ekrana nakon neriješenog rezultata
- *winnerView* – izgled ekrana nakon pobjede jednog igrača
- *update* – osvježava sliku nakon što kliknemo na neko polje
- *insert* – ubacuje novu figuricu u matricu
- *bot* – simulira igranje bota tako što nasumično odabere polje na koje stavlja figuricu
- *size* – pomoćna funkcija korištena u debuggiranju koja vraća veličinu matrice

```
int pressedColumn(int x1, int y1) {
    if (y1 >= 43 && y1 <= 212) {
        if (x1 >= 13 && x1 <= 17) return 1;
        else if (x1 >= 28 && x1 <= 32) return 2;
        else if (x1 >= 43 && x1 <= 47) return 3;
        else if (x1 >= 58 && x1 <= 62) return 4;
        else if (x1 >= 73 && x1 <= 77) return 5;
        else if (x1 >= 88 && x1 <= 92) return 6;
        else if (x1 >= 103 && x1 <= 107) return 7;
        else if (x1 >= 118 && x1 <= 122) return 8;
        else if (x1 >= 133 && x1 <= 137) return 9;
        else if (x1 >= 148 && x1 <= 152) return 10;
        else if (x1 >= 163 && x1 <= 167) return 11;
        else if (x1 >= 178 && x1 <= 182) return 12;
        else if (x1 >= 193 && x1 <= 197) return 13;
        else if (x1 >= 208 && x1 <= 212) return 14;
        else if (x1 >= 223 && x1 <= 227) return 15;
    }
    return -1;
}
```

```
int pressedRow(int x1, int y1) {
    if (y1 >= 43 && y1 <= 212) {
        if (y1 >= 43 && y1 <= 47) return 1;
        else if (y1 >= 58 && y1 <= 62) return 2;
        else if (y1 >= 73 && y1 <= 77) return 3;
        else if (y1 >= 88 && y1 <= 92) return 4;
        else if (y1 >= 103 && y1 <= 107) return 5;
        else if (y1 >= 118 && y1 <= 122) return 6;
        else if (y1 >= 133 && y1 <= 137) return 7;
        else if (y1 >= 148 && y1 <= 152) return 8;
        else if (y1 >= 163 && y1 <= 167) return 9;
        else if (y1 >= 178 && y1 <= 182) return 10;
        else if (y1 >= 193 && y1 <= 197) return 11;
        else if (y1 >= 208 && y1 <= 212) return 12;
    }
    return -1;
}
```

pressedColumn i pressedRow funkcije signaliziraju da li smo pritisnuli na mjesto na kojem želimo da bude figura

```
class Field {
    int X, Y;
    State fieldState;
public:
    Field(int x, int y, State state) {
        X = x;
        Y = y;
        fieldState = state;
    }
    State getState() { return fieldState; }
    void changeState(State state) { fieldState = state; }
    int getX() { return X; }
    int getY() { return Y; }
};
```

Klasa koja označava jednu figuru, opisuje je X i Y koordinatama, te trenutnim stanjem koje može biti slobodno, zauzeto crnom figurom ili zauzeto bijelom figurom.